UNIVERSIDAD CARLOS III DE MADRID
ESCUELA POLITÉCNICA SUPERIOR
GRADO EN INGENIERÍA INFORMÁTICA

CUATROLA PRO: DISEÑO Y DESARROLLO DE UN JUEGO DE CARTAS PARA ANDROID

David Marchante Cano
Tutor: Javier Fernández Muñoz
Octubre de 2014

Universidad Carlos III de Madrid
www.uc3m.es

Título: CUATROLA PRO: Diseño y desarrollo de un juego de cartas para Android.

Autor: David Marchante Cano.

Tutor: Javier Fernández Muñoz.

EL TRIBUNAL

Presidente: _____

Vocal: _____

Secretario: _____

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día __ de _____ de 20__ en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO PRESIDENTE

Resumen

Este proyecto describe el desarrollo de una aplicación para dispositivos Android que implementará toda la lógica necesaria para poder jugar al juego de cartas llamado Cuatrola. La aplicación será desarrollada en modo fuera de línea, por lo que el usuario será un jugador de la partida y, las elecciones de los otros tres participantes, serán llevadas a cabo por la aplicación. La aplicación consta de tres niveles de dificultad en los que se hace uso de algoritmos para la elección de las cartas que irán seleccionando los participantes de la aplicación para ser lanzadas al tapete de la partida. También hace uso de una base de datos para guardar el estado, los puntos y las jugadas de cada partida para generar las estadísticas de cada jugador de la aplicación. El presente documento tiene como finalidad la presentación del proceso completo llevado a cabo para la realización del proyecto.

Abstract

This project describes the development of an application for Android devices that implements the logic needed to play the card game called Cuatrola. The application will be developed in mode offline, the user will be a player of the game, and the election of the other three participants, will be performed by the application. The application has three difficulty levels that make use of algorithms for choosing the cards that participants will select to be launched on the mat of the round. It also makes use of a database to save the state, points and plays each round to generate the statistics for each player of the application. This document is intended for the presentation of the complete process carried out for the project.

Índice de contenidos

Índice de ilustraciones

Índice de tablas

Glosario

Catálogo de términos y acrónimos específicos del contexto del trabajo.

WAP:	*Wireless Application Protocol* (Protocolo de aplicaciones inalámbricas). Es un estándar abierto internacional para aplicaciones que utilizan las comunicaciones inalámbricas.
IDE:	*Integrated Development Environment* (Entorno de Desarrollo Integrado). Software compuesto de un conjunto de herramientas de programación que permite al programador crear aplicaciones a partir de un lenguaje de programación.
***Plug-in*:**	Es un complemento que se relaciona con una aplicación para aportarle una nueva funcionalidad
SDK:	*Software Development kit* (Kit de Desarrollo de Software). Conjunto de herramientas de desarrollo de software que permiten al programador crear aplicaciones para un sistema concreto.
Android:	Sistema operativo basado en el kernel de Linux diseñado principalmente para dispositivos móviles con pantalla táctil, relojes inteligentes, televisores y automóviles.
Interfaz:	Es el medio con que el usuario puede comunicarse con una máquina, un equipo o una computadora, y comprende todos los puntos de contacto entre el usuario y el equipo.
MVC:	Patrón arquitectónico que define la organización independiente del Modelo (Objetos de Negocio), la Vista (interfaz con el usuario u otro sistema) y el Controlador (controlador del flujo de control de la aplicación).
Prototipo:	Primer ejemplar de alguna cosa que se toma como modelo para crear otros de la misma clase.

Chapter 1

Introduction and aims

In this chapter are discussed the objectives and motivation that led to the development of the project, presenting an overview of it and entering the development process from the earliest stage. To facilitate the task of the reader, also there is a section dedicated to briefly comment on the content of each chapter, a summary of each.

1.1. Overview

The purpose of this work is to explain the development of an application for a user to be able to play the game called Cuatrola in offline mode. This means that the user accessing the application will play with a partner and two opponents that will be handled by the application when exists any choice. Possible choices that participants will take regarding the application are the choice of whether a participant play a special game (Solo, Cuatrola or Quintola) and the choice of cards that participants will launch in each one of its turns are played.

The application will adjust to the different types of existing rules so that a user can choose to play with the rules of the different Autonomous Community (Madrid and Extremadura).

The application can also register as many users as needed when their names are always different. Similarly, these may change the names of the three participants with the player playing as long as neither of them are the same to avoid confusion about which participant does what.

They have a feedback window that the player can consult along each hand in case you missed some relevant information from the game.

It may choose to play with three different difficulty levels (easy, medium and hard). Depending on the difficulty level chosen, the application will choose which card will be launched by each participant in his turn differently.

The application will have the option to display, for each player, the statistics that he have had in each of the three levels. For each level will display:

- The number of games played, how many wins, how many lost and the percentage of these.
    - - The total number of points played, how many wins, how many lost and the percentage of these.
    - - The number of Solos that has thrown the player, how many wins, how many lost and the percentage of these.
    - - The number of Cuatrolas that has thrown the player, how many wins, how many losses and the percentage of cells.
    - - The number of Quintolas that has thrown the player, how many wins, how many losses and the percentage of cells
- The number of points total played, how many wins, how many lost and the percentage of these.
- The number of Solos that has played the player, how many wins, how many lost and the percentage of these.
- The number of Cuatrolas that has played the player, how many wins, how many lost and the percentage of these.

- The number of Quintolas that has played the player, how many wins, how many lost and the percentage of these.

If a round is paused or out of it with no finish, when the player goes to access, he will be asked if he wants to resume the round that had started. If he chooses yes, the round will resume again starting the last hand of the game was abandoned. If you choose no, the round will lost and a new game will begin.

For counting statistics, remember the name of the players and participants, and can resume an existing round, it will make use of a database for the application.

1.2. Motivation

Today, practically every existing cards plays have one or more applications to be played from any Smartphone.

Because Cuatrola is not a game spread throughout the national territory, only has an application in Android devices in order to play this game. This application today has 50,000 downloads and 3.8 note out of 5 based on 411 comments. The vast majority of these comments appreciate positively the application by the level of difficulty of implementation and the proper use of animations and the appearance of the user interface, but all negative comments critical application functionality in the real game not perform as well (will be discussed later).

With these data, the motivation of this project is to introduce competition to the existing application, improving everything possible. To do this, a survey of users comments will be made, and the new application will be implemented with facets praised similar to the existing application by users and will improve or create the functionality that the application receives negative comments.

1.3. Aims

El objetivo principal de este proyecto es el desarrollo e implementación de una aplicación móvil para que un usuario sea capaz de poder jugar al juego de la Cuatrola.

The main objective of this project is the development and implementation of a mobile application for a user to be able to play the game of Cuatrola.

This overall goal includes other objectives, which are given below and allow the development of the prototype of the application for the full achievement of the main objective of the project:

- Application development to play with the rules of the desired autonomous region
- Development of the ability to choose between three different difficulty
- Choose whether the player wants partner can play special plays.

Para poder conseguir estos objetivos, se plantea una serie de sub-objetivos más específicos que atienden a los requisitos de cada caso, como son

To achieve these objectives, a number of more specific sub-objectives arise of the requirements, such as:

- The interface should be intuitive and easy to use to improve the user experience.
- It must be the player the option to view the statistics for each difficulty level.
- It will not be permitted to shoot any card that is not permitted to any participant or the player of the round. That is, will not be allowed to make any refuse in the application.

1.4. Developmental stages

Project development can be summarized in six phases.

The first of these phases coincides with the study of the technologies, platforms and existing products, which will carry out a feasibility study of the system. Based on the information obtained at this stage the platform to be used in the project is decided and the architecture and features of the platform are analysed.

Once become comfortable with the platform, in phase two performs the analysis of the requirements that the system must satisfy. For this, will make a study of the application already exists on Android of Cuatrola based on the comments that exist to improve the negative reviews on the implementation is done.

Thus, it is easy to obtain the requirements through the analysis of this information.

In phase three, taking the requirements, it is necessary to design the architecture of the system establishing the components that cover each requirement. What this allows is to ensure that all needs contained in requirements are cover in the project.

At this point you have all the necessary information to start programming, so stage four corresponds to the implementation of code.

Once that the part of the implementation phase finishes, begins the stage of testing through a set of tests, so that it refines the operation of the system, fixing bugs and shaping behaviour.

Finally, the sixth phase is the preparation of the current document.

1.5. Document structure

This section is intended to briefly explain the points developed in the different chapters of the document

Chapter 1.

It aims to provide an overview of the project and presents the objectives and motivations.

Chapter 2.

The evolution of technology in different fields was analyzed and, based on this and future expectations, the choice of development platform and technologies for the development of the application are done.

Chapter 3.

This chapter describes the process of requirements analysis showing the needs that the system need to solve, the design decisions for the development and implementation of interfaces and main features briefly code.

Chapter 4.

Detalla la planificación que se quiere seguir a lo largo de las distintas fases del proyecto, así como el presupuesto del sistema completo de acuerdo con la duración estimada.

This chapter details planning that want to follow along the various phases of the project. It also details the budget of the entire system according to the estimated duration and the time payback.

Chapter 5.

This chapter will conduct a study of the regulatory environment in which the application is included Cuatrola Pro.

Chapter 6.

This chapter treats, subjectively, the conclusions arrived once completed the project, as well as future research and development will continue to improve and refine the functionality of the current system.

Capítulo 2.

Estado de la cuestión

Este capítulo tiene como objetivo analizar las tendencias y avances tecnológicos para determinar el mejor enfoque posible para el proyecto. Solo haciendo un análisis del entorno actual, las tecnologías emergentes y los productos existentes en el mercado, será posible crear un software innovador y atractivo al usuario, capaz de cubrir las carencias de los productos actuales.

2.1. Evolución de la tecnología

2.1.1. Evolución de la tecnología en los juegos para dispositivos móviles

Hoy en día jugar en los teléfonos móviles no es ninguna novedad. Un mercado casi dominado por jugadores casuales, niños y jugadores veteranos. Parece que fue ayer cuando comenzábamos a usar teléfonos móviles para algo más que llamar. Cuando comenzamos a enviarnos mensajes de texto, a presumir de pantalla de más de dos líneas, de sonido polifónico, de diferentes tonos de color para la luz de la pantalla o de quien tenía la carcasa más bonita.

Lo de jugar en el móvil no comenzó hace mucho, pero ha evolucionado tan rápidamente que parecen que fue hace siglos. Todo comenzó a mediados de los noventa, en el año 1994, con el Hagenuk MT-2000 y el primer juego de móvil fue el Tetris. Estuvo lejos de ser una revolución y quedó en una simple anécdota para los poseedores de este modelo. No fue hasta 1997 cuando Nokia, que era el referente en tecnologías móviles por aquella época, sacó al mercado el Nokia 6110 que poseía en memoria el juego Snake que revolucionó el mercado.

Ilustración 1. Hagenuk MT-2000

Ilustración 2. Nokia 6110 con juego Snake

Entonces comenzó la revolución. Cada Nokia traía el Snake al que se fueron uniendo otros. Las demás compañías comenzaron a hacer lo propio, incluyendo la pequeña curiosidad de jugar con el teléfono en los ratos muertos. Los gráficos eran bien simples, una pantalla monocroma e iluminada era la tónica por la época.

Al poco tiempo, en el año 2000, Nokia lanzó el 3310 y con él, el poder jugar en el móvil fue aún más popular. El nuevo milenio se abría al mundo electrónico con miedo pero con increíbles y rápidos avances. Con la llegada del WAP se consiguió conectar los teléfonos a internet. El primer teléfono que integró WAP fue el Nokia 8110.

La llegada de WAP propició la llegada de Java a los teléfonos y con él comenzaron a llegar bastantes juegos algo más elaborados. Teléfonos como el Nokia 3410 ya soportaban Java. Juegos arcade como Asteroids o Space Invaders comenzaron a invadir los terminales telefónicos. Portar este tipo de juegos a pantallas monocromáticas era simple y efectivo. Los teléfonos tenían la potencia suficiente como para reproducir ese tipo de gráficos.

Ilustración 3. Nokia 8110

Ilustración 4. Nokia 3410

En 2003, Nokia lanzó el N-Gage, mediante el cual decidió unir una videoconsola portátil con un teléfono. Venía equipado con el sistema operativo Symbian 6.1, pero no era compatible con pantallas horizontales, por lo que la pantalla estaba en posición vertical. El diseño también era malo, para insertar los juegos había que sacar la batería y el auricular y el altavoz estaban en la parte superior del teléfono, lo que hacía que hubiese que colocar el teléfono de canto para poder hablar y esto no era ni estético ni práctico.

Debido a estos problemas, seis meses después Nokia lanzó el N-Gage QD, el cual resolvía los problemas estéticos del N-Gage y los juegos se introducían por una ranura exterior, de esta manera ya no había que estar sacando la batería con cada juego. A estas alturas los juegos de móviles ya sobrepasaban los gráficos de las consolas de 16 bits. Ya se jugaba, se leían emails y se navegaba por internet con el navegador Opera.

Desde 2003 a 2006 los juegos móviles fueron evolucionando rápidamente. En unos pocos años, los móviles habían sobrepasado ampliamente las capacidades gráficas que ofrecían los juegos de Game Boy. Otros teléfonos podían reproducir juegos similares a los vistos en N-Gage y sin formato físico. Los grandes desarrolladores de juegos no tardaron en hacer versiones móviles de grandes títulos de la época como, Driver 3, Vrally o juegos basados en películas del momento.

Algunos teléfonos ya comenzaban a mostrar gráficos poligonales. Teléfonos como el Nokia N80 ya podía mover algunos juegos 3D.

Todo esto cambió con la aparición en 2007 del primer Iphone de Apple con el sistema operativo iOS y el sistema operativo Android en su versión beta. Ya no sólo eran teléfonos, eran unos micro ordenadores o teléfonos inteligentes, con una tienda virtual propia donde comprar aplicaciones y juegos. Llegaron los smartphones.

Ilustración 5. Iphone de Apple

Eran sistemas operativos que podían llegar a mover gráficos poligonales con soltura. Surgieron entonces juegos como Angry Birds que captaron la atención de todos los jugadores casuales.

Hoy en día los juegos en Smartphones son el pan de cada día. Algo que nació como una simple curiosidad, hoy mueve miles de millones de dólares diarios y es cuna y lanzadera de desarrolladores independientes. Prácticamente todo el mundo lleva un teléfono en el bolsillo y un gran porcentaje tiene algún juego instalado o los usa regularmente. El mercado de juegos en dispositivos móviles comienza a agregar poco a poco el mercado de juegos de consolas portátiles, gracias a sus precios, variedad y cantidad. Esto no implica que siempre sean juegos de calidad tanto gráfica como narrativa, durabilidad y jugabilidad.

2.1.2. Evolución de la tecnología en las plataformas móviles

Hoy en día, existe a disposición del consumidor una amplia variedad de dispositivos móviles y marcas que los comercializan. Actualmente, las plataformas más extendidas entre los usuarios de dispositivos móviles y desarrolladores de aplicaciones son Android, iOS y Windows Phone. Para conocerlas, a continuación se describe brevemente cada una de ellas:

Android

Es el sistema operativo de Google. Está basado en Linux y su uso se enfoca a smartphones y tablets. Dispone de una gran comunidad de desarrolladores de aplicaciones, las cuales pueden ser adquiridas a través del mercado Google Play directamente desde los dispositivos.

La programación de las aplicaciones se basa en Java. Desde su comienzo hasta el día de hoy, se han lanzado 4 versiones estables de este sistema operativo, siendo la más reciente la versión 4.4 (conocida como KitKat). No todas las versiones están disponibles para todos los dispositivos y compañías, y se depende del fabricante que da soporte al terminal para poder instalar versiones más recientes.

En Agosto de 2014 el número de aplicaciones en Google Play rondaba 1.300.000 unidades, superando en número de aplicaciones a la tienda de iOS App Store[3].

El primer dispositivo con Android vio la luz en Octubre de 2008 y a finales de 2010 ya era líder del mercado en plataformas para Smartphone, desbancando a Symbian que había mantenido el liderazgo durante años. A nivel mundial, en el segundo cuatrimestre de 2012 alcanzó una cuota de mercado del 68%. En 2013 se alcanzaron casi mil millones de dispositivos en el mundo activos que hacen uso de Android y se calcula que en 2014 se llegarán a los dos mil millones [4].

iOS

Es el sistema operativo desarrollado y distribuido por Apple. Fue lanzado para iPhone y iPod Touch en 2007 y ya podemos encontrarlo en otros dispositivos como iPad y Apple TV.

A diferencia de Windows Phone de Microsoft y Android de Google, Apple no permite la instalación de iOS en hardware que no pertenezca a Apple.

Dispone de tienda Virtual, Apple Store, dnode hay alrededor de 1.200.000 aplicaciones para iOS. En Junio de 2014, la versión más reciente es la 7.1.2, pero se está trabajando en la versión 8 que desbancará a la versión 7.1.2 [5].

Windows Phone

Es el sistema operativo desarrollado por Microsoft. Es el sucesor de la plataforma Windows Mobile, pero a pesar de ello, son incompatibles y smartphones que han funcionado con Windows Mobile hasta ahora, no pueden ser actualizados a Windows Phone. Del mismo modo, Windows Phone no ofrece compatibilidad con aplicaciones de Windows Mobile.

A diferencia de Windows Mobile, que estaba orientado a un mercado profesional, Windows Phone está enfocado a un mercado mucho más comercial, apto para el uso día a día de cualquier tipo de usuario. Windows Phone Store es el market virtual de esta plataforma y en ella se almacenan alrededor de 300.000 aplicaciones para Windows Phone. La versión más reciente de este sistema operativo es Windows Phone 8.1, que se ha dado a conocer en 2014[6].

2.2. Revisión de tecnologías móviles

Tal y como se define en el capítulo de introducción, el objetivo del presente proyecto es el de la implementación de una aplicación basada en un juego de cartas llamada Cuatrola. De acuerdo a esto, a lo largo de esta sección se realiza una revisión de distintas tecnologías móviles. En particular, se han escogido las siguientes: Android, iOS y Windows Phone. Como criterios definidos para su revisión se establecen su relevancia en el mercado actual de smartphones y tablets.

A continuación, se definen los criterios de estudio que servirán para realizar la revisión teórica, así como su relevancia en el proceso de elección de una de las tecnologías. Seguidamente, se presentará la información más importante de cada criterio para cada tecnología en una tabla comparativa. A partir de esta información, se realizará una comparativa textual de las tecnologías, definiendo los pros y contras de cada una de ellas. Finalmente, se realizará una comparación ponderada de las tres tecnologías y se escogerá una de ellas para el desarrollo de la aplicación móvil de “Cuatrola Pro”.

2.2.1. Criterios de evaluación

Antes de comenzar la revisión teórica de las tecnologías, es necesario establecer unos criterios o características relevantes que serán analizados para cada tecnología, y que permitirán realizar posteriormente una comparación de las tecnologías. Los criterios que se analizarán para cada una de las tecnologías son los siguientes:

Curva de aprendizaje:	Grado de dificultad a la hora de aprender la tecnología y porqué. Este criterio cuenta con un gran componente personal y se considera de gran importancia para escoger una tecnología, ya que se va a desarrollar un proyecto en un tiempo limitado.
Coste:	Coste de uso y desarrollo de la tecnología. Al tratarse de un proyecto universitario, también se considera de gran importancia este criterio.
Comunidad de usuarios y desarrolladores:	Alcance de la tecnología a partir del número de usuarios y desarrolladores. Se considera un criterio de gran importancia en la comparación de tecnologías, ya que el desarrollo de una aplicación puede llevar implícita su salida al mercado.
Rendimiento:	Análisis de velocidad, capacidad de procesamiento, fluidez, tiempos de ejecución…
Documentación:	Cantidad de información disponible sobre la tecnología y el lenguaje de programación. Se considera un criterio de gran importancia en la comparación de tecnologías, ya que desarrollar una aplicación para una tecnología poco documentada implica una

dedicación de tiempo mucho mayor que para una tecnología para la que existe mucha documentación.

A continuación, y antes de analizar cada uno de los criterios definidos anteriormente, se presenta en la *Tabla 1* el resumen del análisis teórico realizado para cada una de las tecnologías propuestas.

	Android	**iOS**	**Windows Phone**
Curva de Aprendizaje	Lenguaje Java. Herramientas para novatos.	Lenguaje propio (Objective-C).	Programación orientada a objetos. C#
Coste	Entorno multiplataforma. Licencia de publicación $25.	Entorno Mac. Licencia de publicación $99 al año.	Entorno Windows. Licencia de publicación $99 al año.
Comunidad de Usuarios	Gran popularidad.	Gran popularidad.	En crecimiento.
Comunidad de Desarrolladores	Muy abundante.	Muy abundante.	En crecimiento. Ayudas de Microsoft para el desarrollo.
Rendimiento	Medio (Máquina virtual de Java).	Excelente.	Bueno.
Documentación	Mucha información.	Mucha información.	Página de Microsoft y en páginas externas.

Tabla 1. Resumen de análisis teórico de las tecnologías

2.2.2. Evaluación cualitativa

Tras presentar en la *Tabla 1* el resumen del análisis teórico de las tecnologías propuestas, se va a realizar una evaluación textual de las características de cada tecnología. Esta evaluación consiste en presentar los pros y contras de cada tecnología y ponderar cada uno de los criterios analizados con el fin de escoger finalmente la tecnología más adecuada.

La popularidad de las tecnologías en auge o con un futuro prometedor, en el que habrá una gran comunidad de usuarios, es un gran incentivo a tener en cuenta. Este criterio lo satisfacen los sistemas operativos Android, iOS y, en menor medida, Windows Phone. Gracias a una gran estrategia de expansión de Windows, en colaboración con otra gran empresa como es Nokia, se puede considerar a Windows Phone como potencial competidor en relación a la comunidad de usuarios de un sistema operativo móvil.

Por otro lado, también se ha considerado **la curva de aprendizaje** como uno de los criterios más importantes. Dado el carácter de este proyecto, supone una restricción el aprender un lenguaje en un corto periodo de tiempo e implementar el prototipo de una aplicación móvil. En este punto, se consideran como potenciales tecnologías Android y Windows Phone. Android utiliza lenguaje Java, del que hay infinidad de información para su desarrollo. Windows Phone, está programado en C#, una

variante de C orientado a objetos, del cual se dispone de mucha información, tutoriales, ejemplos, etc. En cuanto a iOS, utiliza un lenguaje único, cerrado y con una gran curva de aprendizaje, sin ninguna similitud a los lenguajes más habituales.

El tercer aspecto y el último que se ha ponderado como una de los más determinantes es el **coste**. La tecnología más interesante en este aspecto es Android. Tanto para Windows Phone como para iOS, la licencia para la publicación de aplicaciones supone $99 anuales. Además, son necesarios dispositivos propios de las marcas para el desarrollo de aplicaciones, es decir, que se necesitará un ordenador *Mac* para programar para iOS y un entorno *Windows* para programar aplicaciones para Windows Phone. Esto encarece aún más la puesta en marcha del desarrollo de aplicaciones para estas plataformas.

La comunidad de desarrolladores arropa una tecnología y le proporciona más vida útil a dicho sistema operativo, lo que garantiza unos años mínimos de funcionamiento. En este criterio pasan la prueba las tres tecnologías. Por otra parte, se encuentra la cantidad de documentación existente para la tecnología, que actualmente está ligada al éxito del sistema operativo y al número de comunidades de usuarios y desarrolladores existentes. De nuevo, las tres tecnologías obtienen una buena nota para este criterio.

Por último, el rendimiento es otra característica importante. Las máquinas virtuales Java no tienen fama de ser rápidas, por lo que desecharemos a Android. Las Windows Phone y iOS se consideran mejores en comparación con las anteriores.

Con el fin de completar la información necesaria para escoger una de las tecnologías, se presentan a continuación los pros y contras de cada una de las tecnologías:

Android

Pros:

- El SDK es multiplataforma.
- El desarrollo es gratuito, y la licencia de desarrollador es vitalicia.
- Facilidad de ejecutar aplicaciones de terceros en un dispositivo Android.
- Facilidad de publicar aplicaciones propias en *Google Play*.

Contras:

- El rendimiento de Java.
- La fragmentación. Existen muchos dispositivos Android distintos, con diferentes resoluciones de pantalla, diferentes procesadores, diferentes versiones del sistema operativo… Esto dificulta la tarea del desarrollador, que debe crear aplicaciones que puedan ser ejecutadas en el mayor número posible de dispositivos.

iOS

Pros:

- La respuesta a las órdenes del usuario es inmediata y provee de una interfaz fluida.
- La interfaz de usuario de iOS está basada en el concepto de manipulación directa, usando gestos multitáctiles.
- Apple lanza sus actualizaciones de manera oficial y casi cada tres meses.
- La gran comunidad de usuarios y desarrolladores de *Apple*.

Contras:

- iOS no permite *Adobe Flash* ni Java, usa HTML5 como una alternativa a *Flash*.
- El precio de los dispositivos con iOS lo limita a un público más determinado.

Windows Phone

Pros:

- Lenguaje orientado a objetos actual: C#. Fácil de leer, mantener y portable.
- Menos código que en otras plataformas para realizar la misma función.
- Se puede prototipar la interfaz gráfica y la aplicación en la misma herramienta.
- La herramienta Expression Blend para el diseño de la interfaz en Windows Phone es una herramienta muy sencilla de utilizar y más potente que en la mayoría de sus competidores en esta evaluación de tecnologías. [7]
- Microsoft ha potenciado al máximo las herramientas de desarrollo de Windows Phone 8.1, lo que se traduce en una programación sencilla y más rápida que para otras plataformas.

Contras:

- Necesario tener sistema operativo Windows Vista o Windows 7.
- Aún no demasiado extendido entre la comunidad de usuarios.
- Aún no disponible en muchos móviles.
- Mercado de aplicaciones aún en crecimiento, sin tantas aplicaciones como en Android o iOS.

2.2.3. Evaluación cuantitativa

	Curva de Aprendizaje	Coste	Comunidad de Usuarios
Android	2	2	2
iOS	0	0	2
Windows Phone	2	0	1

Tabla 2. Comparación ponderada de las tecnologías

Los criterios más relevantes para la elección de la tecnología móvil han sido la curva de aprendizaje, el coste y la comunidad de usuarios. De acuerdo a esto, a cada tecnología se le ha asignado, para cada uno de los criterios, una puntuación entre 0 y 2. Una puntuación de 2 en un criterio dado supone que la tecnología satisface completamente las necesidades del proyecto. De igual manera, una puntuación de 0 en un criterio dado supone que la tecnología no satisface las necesidades del proyecto. La puntuación de 1 se reserva para la satisfacción parcial de las necesidades del proyecto. En base a ello, la *Tabla 2* despliega la puntuación asignada a cada tecnología para cada criterio.

De acuerdo a la *Tabla 2* se puede observar que Android es la única tecnología que satisface completamente los criterios que se han considerado más relevantes. Cabe destacar que debido a las características del presente proyecto no se ha considerado el coste de publicación de aplicaciones Android como un coste significativo, por lo que se ha establecido la mayor puntuación para este criterio. Con todo, Android se escoge como la tecnología sobre la que se implementará la aplicación móvil “Cuatrola Pro”.

2.3. Estudio de alternativas de la solución

En este apartado se va a llevar a cabo un estudio de todas las tecnologías y herramientas que van a ser utilizadas en el proyecto. Se presentaran distintas alternativas para, posteriormente, clasificarlas y realizar una elección basada en el estudio y evaluación. A continuación se presentaran diversas alternativas agrupados en los diferentes sectores.

2.3.1. Tecnologías de desarrollo

- **Java + SDK**: Conjunto de herramientas, documentación, ejemplos de código y tutoriales diseñado principalmente para desarrollar aplicaciones para el Sistema Operativo Android.
- **jQuery Mobile:** Marco de trabajo basado en jQuery y HTML5 que nos permite generar aplicaciones móviles para cualquier dispositivo.

2.3.2. Entornos de desarrollo

Para el desarrollo de la aplicación es necesario trabajar sobre un entorno que ofrezca numerosas herramientas para que sea una tarea lo más sencilla posible para el desarrollador. A continuación se presentan varias alternativas en función de la tecnología de desarrollo.

Para Java + SDK

- **Android Studio:** Entorno de Desarrollo Integrado (IDE) para la plataforma Android presentado por Google en 2013. Se trata de una plataforma basada en el IDE de Java IntelliJ IDEA. Una de sus principales características es que permite pre visualizar el diseño de la aplicación.
- **Eclipse + ADT plugin:** Eclipse es un entorno de desarrollo integrado con gran aceptación entre los desarrolladores de aplicaciones. Una de las principales características de este entorno de desarrollo es la cantidad de extensiones que existen para desarrollar en diversos lenguajes y con numerosas tecnologías. Una de estas extensiones es el ADT que es el entorno más popular y utilizado para desarrollo en Android.

Para JQuery Mobile

- **Eclipse + librería jQuery:** Entorno de desarrollo Eclipse con la librería jQuery importada. Esto permite desarrollar aplicaciones en marcos móviles.
- **Eclipse + jQuery Mobile plugin:** Consiste en el mismo entorno de desarrollo detallado en el punto anterior de Eclipse + ADT plugin, pero con otra extensión: jQuery Mobile plugin. Esta es

la práctica más popular y sencilla para el desarrollador ya que incluye numerosas herramientas para simplificar el desarrollo.

2.3.3. Alternativas para sistema gestor de bases de datos

Dado que la aplicación va a ser implementada en un sistema móvil (en este caso Android) se utilizara **SQLite** para la parte de cliente ya que no necesitamos un sistema de gestión completo, sino un sistema de gestión muy ligero que realice operaciones de la manera más rápida y eficiente posible.

2.3.4. Valoración de las alternativas

Para poder valorar las alternativas propuestas, a continuación se definen los conceptos a utilizar como criterio a la hora de evaluar cada alternativa. De esta manera podemos determinar la opción que mejor se adapta al sistema a implantar.

Para cada elemento se muestra una tabla en la que, para cada concepto, se muestra la relevancia de éste, la puntuación y la valoración ponderada. Además se calcula un total que consiste en la media aritmética de las valoraciones de todos los componentes.

Los conceptos que se van a tener en cuenta para calcular la valoración son los siguientes:

- **Documentación:** Este concepto hace referencia a la cantidad y calidad de la documentación de un determinado elemento (manuales, tutoriales, bibliográfica, artículos en revistas, etc.). En las tablas siguientes se puntúa con 0 la ausencia de documentación y/o materiales de referencia y con 10 la existencia de documentación abundante y de fácil acceso.
- **Estabilidad:** Con este concepto se mide la capacidad del sistema para funcionar sin imprevistos o errores inesperados que puedan poner el peligro el funcionamiento del sistema. Se considera con la puntuación de 0 que el sistema es completamente inestable y con 10 un sistema absolutamente fiable.
- **Formación:** Mediante este concepto se mide la formación que será necesaria impartir al equipo para alcanzar los conocimientos suficientes para interactuar de forma eficiente y eficaz con el elemento descrito. Se puntúa con 0 la necesidad absoluta de formación y con 10 que no es necesario impartir formación.
- **Impacto:** Este término mide el grado de dificultad que supondrá al equipo elaborar el sistema en base a los conocimientos que conocen de antemano. Se puntúa con 0 el mayor impacto y con 10 la ausencia del mismo.
- **Mantenimiento**: Con este concepto se mide el grado que ofrece el elemento para llevar a cabo el mantenimiento o la modificación del sistema para adaptarlo a nuevas situaciones o requerimientos. Se puntúa con 0 la falta de versatilidad del elemento y con 10 una alta capacidad de adaptación.
- **Soporte:** Este concepto es utilizado para valorar si el elemento dispone de algún tipo de soporte o asistencia técnica por parte del proveedor, distribuidor o fabricante. Se utiliza una puntuación de 0 para indicar la ausencia de soporte y con 10 la existencia de un soporte de calidad.

- **Precio:** Es utilizado para medir la inversión económica necesaria para realizar la compra del elemento evaluado. Se valora con 0 en caso de que el elemento no suponga ningún coste y con 10 en caso de que el elemento tenga un precio muy elevado en relación a los valores del mercado.

Valoración cuantitativa de las alternativas de desarrollo

Java + SDK							
Android Estudio				**Eclipse + ADT Plugin**			
Concepto	Relevancia	Puntuación	Valoración Ponderada	Concepto	Relevancia	Puntuación	Valoración Ponderada
Documentación	8	7	56	Documentación	8	9	72
Estabilidad	7	8	56	Estabilidad	7	8	56
Formación	6	6	36	Formación	6	6	36
Impacto	7	5	35	Impacto	7	4	28
Mantenimiento	7	7	49	Mantenimiento	7	8	56
Soporte	6	7	42	Soporte	6	7	42
Precio	5	2	10	Precio	5	3	15
Total (Sobre 10)			**4,06**	**Total (Sobre 10)**			**4,36**

Tabla 3. Ponderación Java + SDK

jQuery Mobile							
Eclipse + Librería jQuery				**Eclipse + jQuery Mobile Plugin**			
Concepto	Relevancia	Puntuación	Valoración Ponderada	Concepto	Relevancia	Puntuación	Valoración Ponderada
Documentación	8	6	48	Documentación	8	6	48
Estabilidad	7	7	49	Estabilidad	7	8	56
Formación	6	8	48	Formación	6	7	42
Impacto	7	6	42	Impacto	7	7	49
Mantenimiento	7	7	49	Mantenimiento	7	8	56
Soporte	6	6	36	Soporte	6	5	30
Precio	5	3	15	Precio	5	2	10
Total (Sobre 10)			**4,10**	**Total (Sobre 10)**			**4,16**

Tabla 4. Ponderación jQuery Mobile

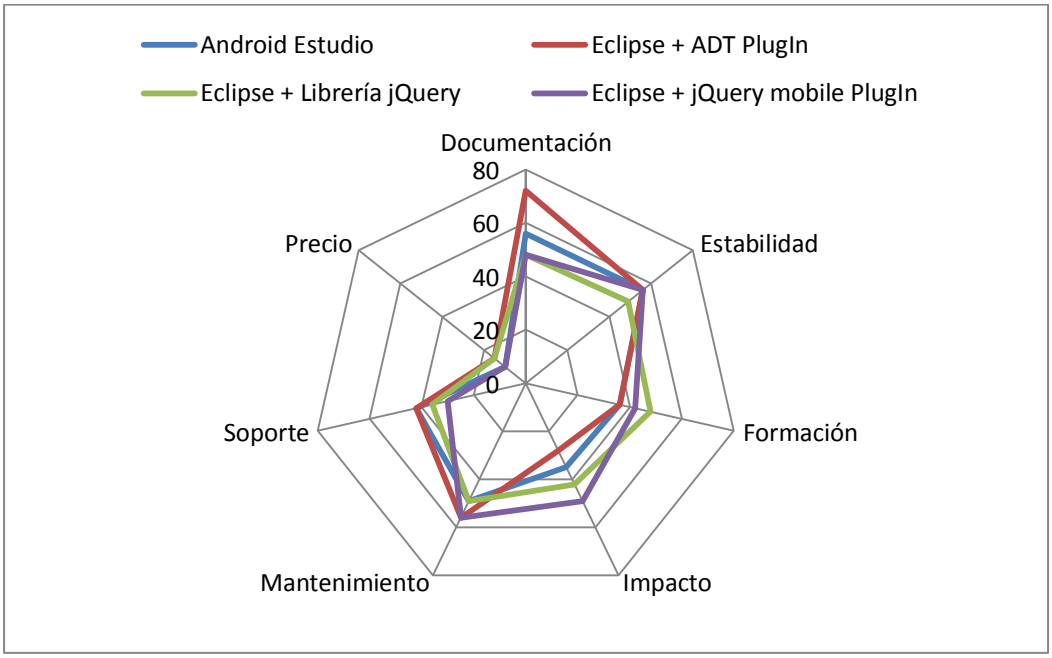

Ilustración 6. Gráfico comparativo de alternativas de desarrollo

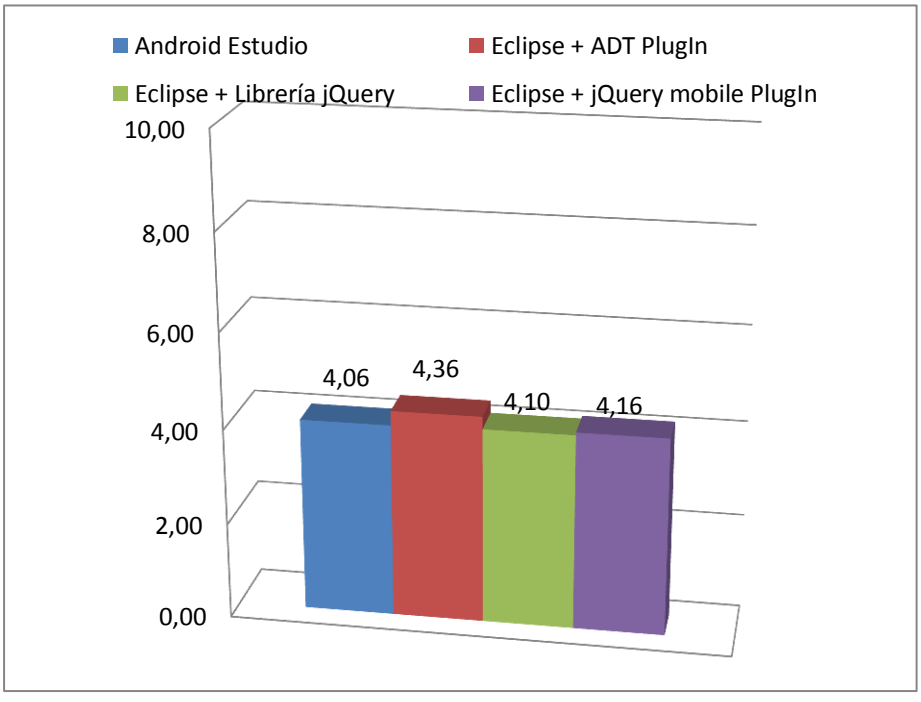

Ilustración 7. Gráfica de totales en alternativas de desarrollo

2.4. Análisis de aplicaciones

Desde la llegada de Android a los smartphones, el número de aplicaciones en el market de Android ha crecido rápidamente. A pesar de esto, debido a que el juego de la Cuatrola únicamente se encuentra extendido con mayor amplitud por zonas de Extremadura y Madrid, el juego de la Cuatrola sólo tiene una aplicación en el market de Android para poder jugar. En cambio, para iOS o Windows Phone no posee ninguna.

En este apartado analizaremos la aplicación encontrada en Google Play llamada Cuatrola, mostrando las ventajas y desventajas de la misma.

Estudio de la aplicación Cuatrola

Esta aplicación permite jugar al juego de la cuatrola. Consta de tres pantallas: una principal, una que explica cómo jugar y la pantalla para jugar. A continuación analizaremos sus ventajas y desventajas.

Ilustración 8. Imagen Aplicación Cuatrola

Ventajas

- Interfaz con buena estética, clara e intuitiva.
- Buen uso de las animaciones de Android.

Desventajas

- Carencia de nombres.
- Selecciona siempre el peor caso posible de carta a lanzar.
- No sabes en qué palo cantas.
- Tu compañero no se juega ninguna jugada.
- Eliges siempre tú primero si te vas a por jugada.
- Las partidas son infinitas.
- Ausencia de base de datos para recuperar partidas.

Ilustración 9. Tapete Aplicación Cuatrola

2.5. Selección de la solución

Tras el estudio realizado de las alternativas para llevar a cabo el proyecto "Cuatrola Pro", se ha tomado la decisión de trabajar con las tecnologías y herramientas que se muestran en la siguiente tabla:

Selección de Tecnologías y Herramientas para el proyecto Cuatrola Pro	
Sistema Operativo	Android 4.4 (KitKat)
Alternativas de desarrollo	Java + SDK → Eclipse + ADT Plugin
Sistema Gestor de Base de Datos	SQLite

Tabla 5. Selección de Tecnologías y Herramientas para el Proyecto Cuatrola Pro

Capítulo 3.

Análisis, Diseño e Implementación

En este capítulo se especifican las fases de análisis de requisitos, diseño de la aplicación e implementación a nivel de código. A final del capítulo se detallan algunas de las pruebas utilizadas para el testeo de la aplicación móvil.

3.1. Análisis

3.1.1. Requisitos de usuario

Se dividirán los requisitos de usuario en dos grandes bloques: requisitos de capacidad y requisitos de restricción. Los requisitos de capacidad serán los requisitos que informan de qué tiene que hacer el sistema y los de restricción de cómo lo hará.

Con el fin de mantener una mejor trazabilidad de los requisitos, dentro de estos dos grandes bloques se realizarán clasificaciones de los requisitos. Para identificar los requisitos se utilizará la siguiente tabla:

XXX - YY			
Nombre			
Prioridad		**Necesidad**	
Verificabilidad		**Claridad**	
Estabilidad			
Descripción			

Tabla 6. Formato tabla requisitos

En donde cada uno de los campos será:

- **Identificador:** el identificador es el que se encuentra en la cabecera de la tabla. Está definido por tres caracteres alfabéticos (XXX) y dos dígitos (YY). Los tres caracteres alfabéticos identificarán el tipo de requisito que es, si es de capacidad los caracteres serán RUC y si son de restricción serán RUR. Los dos dígitos identificarán el número de requisito dentro de cada clasificación.
- **Nombre:** nombre del requisito.
- **Prioridad:** nivel de importancia del requisito. Sus valores serán Alta, Media, Baja según el nivel de importancia que tenga el requisito.
- **Verificabilidad:** indicará el nivel de pruebas que se pueden llegar a realizar para probar el requisito. Sus valores serán Alta, Media y Baja.
- **Estabilidad:** representa durante qué período del proyecto el requisito será estable.
- **Necesidad:** identifica el nivel en el que el requisito puede ser modificado. Sus valores serán:
    - Esencial: el requisito no puede modificarse.

- Deseable: se puede modificar.
- Opcional: se puede modificar y eliminar.

- **Claridad:** mide la ambigüedad de cada uno de los requisitos. Sus valores serán Alta, Media y Baja según la ambigüedad del requisito. Donde Alta significará una gran claridad y Baja una gran ambigüedad.
- **Descripción:** proporcionará una definición detallada del requisito.

3.1.1.1. Requisitos de capacidad

RUC - 01			
Nombre	Visualizar Reglas		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Cualquier usuario, ya esté identificado por el sistema o no, podrá visualizar las reglas del juego Cuatrola.		

Tabla 7. RUC - 01

RUC - 02			
Nombre	Crear Jugador		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Un usuario no identificado podrá crear un nuevo jugador en la aplicación cada vez que lo desee.		

Tabla 8. RUC – 02

RUC - 03			
Nombre	Identificarse		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Un usuario no identificado podrá acceder a la aplicación como un usuario ya creado anteriormente.		

Tabla 9. RUC - 03

RUC - 04			
Nombre	Reanudar Partida		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Cuando un usuario se identifica, tendrá la opción de reanudar la última partida jugada si no la acabó.		

Tabla 10. RUC - 04

RUC – 05			
Nombre	Seleccionar Comunidad Tanteo		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Un usuario identificado tendrá la posibilidad de elegir si usar un tanteo basado en las reglas de la comunidad autónoma de Madrid o Extremadura.		

Tabla 11. RUC - 05

RUC - 06			
Nombre	Seleccionar Nivel Dificultad		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Un usuario identificado tendrá la posibilidad de elegir jugar una partida en diferentes niveles de dificultad.		

Tabla 12. RUC - 06

RUC - 07			
Nombre	Elegir Compañero Solo		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Un usuario identificado tendrá la posibilidad de elegir si quiere que su compañero pueda jugarse jugadas especiales sin que juegue él o si no.		

Tabla 13. RUC - 07

RUC - 08			
Nombre	Cambiar Nombres Participantes		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Un usuario identificado tendrá la posibilidad de cambiar el nombre de los tres participantes de su partida.		

Tabla 14. RUC - 08

RUC - 09			
Nombre	Salir		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Un jugador podrá salir de la aplicación en cualquier punto de ella.		

Tabla 15. RUC - 09

RUC - 10			
Nombre	Abandonar Partida		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Un jugador podrá abandonar una partida mientras la esté jugando, dándola por perdida.		

Tabla 16. RUC - 10

RUC – 11			
Nombre	Cantar		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Si un jugador posee en la mano el rey y el caballo del mismo palo, podrá cantar en dicho palo.		

Tabla 17. RUC - 11

RUC - 12			
Nombre	Tirar Carta		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Cada vez que sea el turno de un jugador en una baza, éste tirará una carta de las que posea en la mano.		

Tabla 18. RUC - 12

RUC - 13			
Nombre	Ordenar Cartas		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Un jugador podrá ordenar en todo momento las cartas que posea en cada mano.		

Tabla 19. RUC - 13

RUC – 14			
Nombre	Ver Puntuación		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Un jugador tendrá constantemente visible la puntuación que se vaya teniendo en todo momento durante una partida.		

Tabla 20. RUC - 14

RUC – 15			
Nombre	Seleccionar Jugadas		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Un jugador decidirá si quiere jugarse una jugada especial o no.		

Tabla 21. RUC - 15

RUC – 16			
Nombre	Ver Pinte		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Un jugador tendrá constantemente visible la carta que sea el pinte en todo momento durante cada mano.		

Tabla 22. RUC - 16

RUC – 17			
Nombre	Visualizar Estadísticas		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Un jugador podrá visualizar las estadísticas de todas las partidas que haya jugado.		

Tabla 23. RUC - 17

3.1.1.2. Requisitos de restricción

RUR - 01			
Nombre	Tamaño Nombres		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Los nombres de todos los participantes de la aplicación tendrán entre 1 y 11 caracteres.		

Tabla 24. RUR - 01

RUR – 02			
Nombre	Nombre Jugador No Repetido		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	No se podrán crear dos jugadores con el mismo nombre.		

Tabla 25. RUR - 02

RUR – 03			
Nombre	Listado nombres jugadores		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Se mostrará un listado con los nombres de todos los jugadores creados en la aplicación.		

Tabla 26. RUR – 03

RUR – 04			
Nombre	Seleccion Jugador		
Prioridad	Alta	**Necesidad**	Deseable
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	A la hora de identificarse, el usuario tendrá que seleccionar el nombre de un jugador existente del listado de nombres de los jugadores.		

Tabla 27. RUR – 04

RUR – 05			
Nombre	Reanudación Partida		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Cuando un usuario se identifique, si tiene una partida sin terminar, se le preguntará si quiere continuar la partida. En caso de seleccionar que no, se dará la partida inacabada por finalizada y perdida		

Tabla 28. RUR – 05

RUR – 06			
Nombre	Momento Reanudación		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Una partida será reanudada repartiendo de nuevo en la última mano en la que fue parada la partida.		

Tabla 29. RUR - 06

RUR – 07			
Nombre	Comunidad Tanteo Por Defecto		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Saldrá seleccionada la opción del tanteo de la comunidad de Madrid por defecto.		

Tabla 30. RUR – 07

RUR – 08			
Nombre	Validación Nivel Dificultad		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Antes de comenzar una nueva partida, se comprobará si el usuario ha seleccionado un nivel de dificultad. Si no lo ha hecho, se le mostrará un mensaje por pantalla y no se permitirá comenzar la partida.		

Tabla 31. RUR – 08

RUR – 09			
Nombre	Niveles Dificultad		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Existirán tres niveles de dificultad posibles: Fácil, Medio y Difícil.		

Tabla 32. RUR – 09

RUR – 10			
Nombre	Compañero Solo Por Defecto		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Aparecerá seleccionado que el compañero se puede ir Solo por defecto.		

Tabla 33. RUR - 10

RUR – 11			
Nombre	Nombres Partida No Repetidos		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	No se permitirá que dentro de una partida haya nombres de algunos de los participantes repetidos.		

Tabla 34. RUR – 11

RUR – 12			
Nombre	Caracteres Nombres		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Para los nombres de la aplicación se permitirá cualquier carácter alfanumérico.		

Tabla 35. RUR – 12

RUR – 13			
Nombre	Posibilidad salir aplicación		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	El usuario podrá salir de la aplicación pulsando el botón de retroceso de su dispositivo móvil en cualquier momento.		

Tabla 36. RUR – 13

RUR – 14			
Nombre	Menú abandonar		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Cuando se solicite abandonar una partida, se mostrará un menú en que se explique que la partida será contabilizada como perdida y se confirme el abandono.		

Tabla 37. RUR – 14

RUR – 15			
Nombre	Visualización Cante		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Se mostrará un listado que se actualizará cada mano dependiendo del palo de la carta pinte para indicar en qué palos se cantan 20 y en qué palo se cantan 40.		

Tabla 38. RUR - 15

RUR – 16			
Nombre	Realización Cante		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Sólo se podrá realizar un cante cuando, el participante que vaya a cantar o su compañero, ganen una baza de la mano.		

Tabla 39. RUR – 16

RUR – 17			
Nombre	Número Cantes		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Sólo se podrá realizar un cante por participante cuando se gane una baza.		

Tabla 40. RUR – 17

RUR – 18			
Nombre	Validación Carta		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	La aplicación sólo permitirá que el usuario tire una carta cuando ésta carta cumpla con las reglas de la Cuatrola a la hora de ser seleccionada.		

Tabla 41. RUR – 18

RUR – 19			
Nombre	Ordenación Cartas		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	La ordenación de las cartas del jugador se realizará por palos, siguiendo el siguiente orden: oros, copas, espadas y bastos. Dentro de cada palo se ordenarán las cartas de mayor a menor puntuación de izquierda a derecha.		

Tabla 42. RUR – 19

RUR – 20			
Nombre	Formato Puntuación		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	La puntuación de la partida se representará mediante números enteros que permanecerán visibles continuamente en la vista de la partida.		

Tabla 43. RUR – 20

RUR – 21			
Nombre	Jugadas Permitidas		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	A la hora de seleccionar las jugadas, el jugador dispondrá entre las siguientes opciones: - **Solo:** el jugador indica que ganará la partida sin su compañero. - **Cuatrola:** el jugador indica que se llevará cuatro bazas sin su		

	compañero. - **Quintola:** el jugador indica que se llevará las cinco bazas sin su compañero. - **Nada:** el jugador indica que no jugará ninguna jugada especial.

Tabla 44. RUR – 21

RUR – 22			
Nombre	Selección Jugada		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Un participante únicamente tendrá la opción de seleccionar una jugada especial si ningún participante anterior ha indicado que se juega alguna jugada especial antes.		

Tabla 45. RUR – 22

RUR – 23			
Nombre	Visualización Pinte		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	El pinte de la partida se visualizará en todo momento durante la mano en la que pinte. Se representará con una imagen de la carta que haya sido el pinte de la mano.		

Tabla 46. RUR – 23

RUR – 24			
Nombre	Representación Pinte Participantes		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Si el repartidor es uno de los participantes que no sea el jugador de la partida, se mostrará la carta pintada dada la vuelta en las cartas de la mano de dicho participante. Cuando el participante tire esta carta al tapete, desaparecerá el pinte de las cartas de su mano.		

Tabla 47. RUR - 24

RUR – 25			
Nombre	Clasificación Estadísticas		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Las estadísticas de cada jugador serán mostradas por niveles de dificultad. Se realizarán las estadísticas para el nivel Fácil, para el nivel Medio y para el nivel Difícil por separado.		

Tabla 48. RUR - 25

RUR – 26			
Nombre	Campos Estadísticas		
Prioridad	Alta	**Necesidad**	Deseable
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Los campos sobre los que se realizarán las estadísticas son: - **Partidas Jugadas:** son el total de partidas jugadas. Dentro de este campo se dividirán por ganadas o perdidas y el porcentaje de partidas ganadas. - **Puntos Totales:** son el total de puntos jugados en todas las partidas. Dentro de este campo se dividirán por el número de ganados o perdidos		

	y el porcentaje de puntos ganados. - **Solos:** son el total de Solos que ha echado el jugador en todas las partidas. Dentro de este campo se dividirán por el número de ganados o perdidos y el porcentaje de Solos ganados. - **Cuatrolas:** son el total de Cuatrolas que ha echado el jugador en todas las partidas. Dentro de este campo se dividirán por el número de ganadas o perdidas y el porcentaje de Cuatrolas ganadas. - **Quintolas:** son el total de Quintolas que ha echado el jugador en todas las partidas. Dentro de este campo se dividirán por el número de ganadas o perdidas y el porcentaje de Quintolas ganadas.

Tabla 49. RUR - 26

3.1.2. Casos de uso

En este apartado se especificarán los casos de uso. Estos son una descripción de los pasos o las actividades que deberán realizarse para llevar a cabo el uso de la aplicación por parte de los usuarios. Los casos de uso del sistema serán representados en el siguiente diagrama:

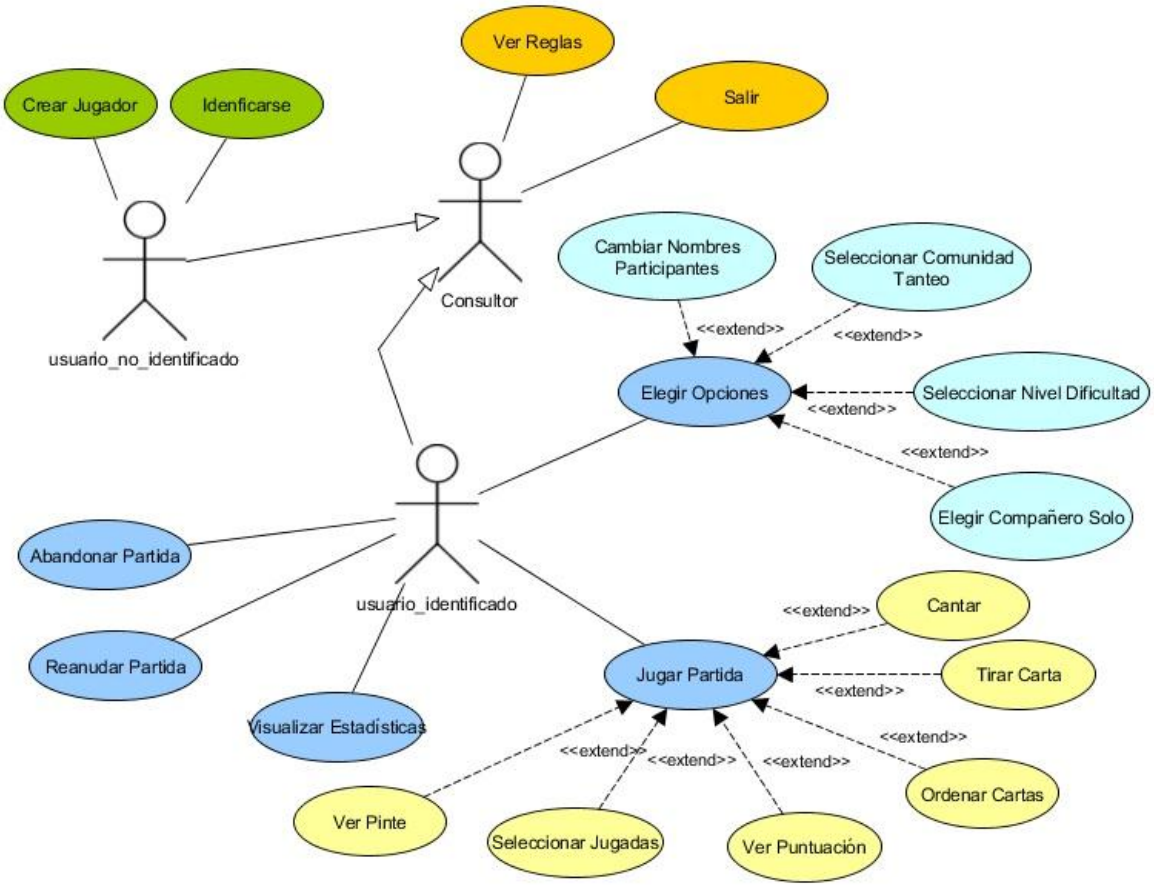

Ilustración 10. Diagrama Casos de Uso

Para facilitar la lectura del diagrama, organizaremos la información en tablas cuyo formato incluirá los siguientes campos:

- **Identificador:** nomenclatura que identifica inequívocamente al caso de uso. Se define como CU-XX, donde XX es un número que representa el caso de uso.
- **Título:** nombre que se le da al caso de uso.
- **Actores:** los posibles roles de usuario que pueden intervenir en el caso de uso.
- **Descripción:** breve explicación del proceso llevado a cabo en el caso de uso.
- **Pre-condiciones:** son aquellos hechos que deben ser previamente ciertos para poder llevar a cabo el caso de uso.

- **Post-condiciones:** Son aquellos hechos que la correcta ejecución del caso de uso los hacen posibles.
- **Escenario:** descripción esquemática de las fases que componen el caso de uso.
- **Condiciones de fallo:** describe posibles errores y las respuestas del sistema ante los mismos.

A continuación se detallan las tablas de los casos de uso con los campos definidos:

CU-01	
Título	Ver Reglas
Actores	Consultor, usuario_no_identificado y usuario_identificado
Descripción	Ver las reglas del juego de la Cuatrola
Precondiciones	Ninguna
Post-condiciones	Aparecerán en pantalla las reglas del juego de la Cuatrola.
Escenario	1. Aparece la pantalla "principal" se pulsa botón Reglas. 2. Se muestra la pantalla "reglas"
Condiciones de fallo	Ninguna

Tabla 50. CU – 01

CU-02	
Título	Salir
Actores	Consultor, usuario_no_identificado y usuario_identificado
Descripción	Ver las reglas del juego de la Cuatrola
Precondiciones	Ninguna
Post-condiciones	Se podrá salir de la aplicación cuando el usuario lo desee.

Escenario	1. Se pulsará botón del dispositivo móvil para salir de la aplicación.
Condiciones de fallo	Ninguna

Tabla 51. CU - 02

CU-03	
Título	Crear Jugador
Actores	usuario_no_identificado
Descripción	Crear un nuevo jugador en la aplicación
Precondiciones	No exista ya un usuario con el mismo nombre creado.
Post-condiciones	Se guardará el nombre del nuevo jugador en la base de datos.
Escenario	1. Aparece la pantalla "crear_jugador". 2. Se escribe el nombre deseado en el cuadro de texto. 3. Se pulsa el botón Crear Jugador.
Condiciones de fallo	Exista jugador con el mismo nombre ya creado.

Tabla 52. CU - 03

CU-04	
Título	Identificarse
Actores	usuario_no_identificado
Descripción	Identificarse como un jugador creado previamente en la aplicación.
Precondiciones	Exista el nombre del jugador ya en la base de datos
Post-condiciones	Se identificará al usuario con el nombre del jugador seleccionado.
Escenario	1. Aparece la pantalla "seleccionar_jugador". 2. Se elige el jugador que se quiere solicitar del listado de jugadores. 3. Se pulsa el botón Seleccionar Jugador

Condiciones de fallo	No exista nombre jugador creado previamente.

Tabla 53. CU - 04

CU-05	
Título	Abandonar Partida
Actores	usuario_identificado
Descripción	Se abandonará la partida marcándose como perdida
Precondiciones	Que el usuario se haya identificado y esté en medio de una partida.
Post-condiciones	Se marcará la partida como perdida y se volverá a la pantalla principal.
Escenario	1. Jugador jugando partida. 2. Aparece pantalla opciones_partida. 3. Pulsa botón Abandonar. 4. Aparece pantalla abandonar_partida. 5. Pulsa botón Si.
Condiciones de fallo	Ninguna

Tabla 54. CU - 05

CU-06	
Título	Reanudar Partida
Actores	usuario_identificado
Descripción	Un jugador puede reanudar una partida empezada e inacabada por haberse salido anteriormente.
Precondiciones	Jugador se haya salido en la última partida jugada de la aplicación cuando la partida no había acabado aún.
Post-condiciones	Se cargarán todos los valores de la partida anterior y se volverá a repartir la mano en la que el jugador abandonó la partida.
Escenario	1. Jugador sale de partida en juego. 2. Usuario entra en aplicación y se identifica con el nombre del jugador que seleccionó la partida

	3. Aparece pantalla “reanudar_partida” y se pulsa botón Si.
Condiciones de fallo	Ninguna

Tabla 55. CU - 06

CU-07	
Título	Visualizar Estadísticas
Actores	usuario_identificado
Descripción	Se mostrarán las estadísticas del jugador seleccionado por nivel de dificultad.
Precondiciones	Que haya algún jugador creado en la base de datos
Post-condiciones	Aparecerán en pantalla las estadísticas del jugador seleccionado.
Escenario	1. Aparece la pantalla “seleccionar_jugador”. 2. Se selecciona el jugador del que se desean ver las estadísticas. 3. Se pulsa botón Estadísticas. 4. Se muestra la pantalla “estadisticas”
Condiciones de fallo	Que no haya ningún usuario creado.

Tabla 56. CU - 07

CU-08	
Título	Elegir Opciones
Actores	usuario_identificado
Descripción	Engloba la elección de todas las opciones de una partida
Precondiciones	Usuario identificado como un jugador ya creado
Post-condiciones	Se seleccionarán las opciones elegidas.
Escenario	1. Se muestra pantalla “seleccionar_opciones”

	2. Se eligen las opciones deseadas de la partida
Condiciones de fallo	Ninguna

Tabla 57. CU - 08

CU-09	
Título	Elegir Compañero Solo
Actores	usuario_identificado
Descripción	Elección de la posibilidad si el compañero del jugador puede jugar jugadas especiales o no.
Precondiciones	Usuario identificado como jugador creado previamente.
Post-condiciones	Se seleccionará si el compañero puede irse a por jugada especial durante la partida a jugar.
Escenario	1. Aparece la pantalla "seleccionar_opciones". 2. Se marca el checkbox del compañero solo si se quiere que pueda irse solo, o no se tica si no se quiere que se vaya solo. 3. Se pulsa botón Empezar.
Condiciones de fallo	Ninguna

Tabla 58. CU - 09

CU-10	
Título	Seleccionar Nivel Dificultad
Actores	usuario_identificado
Descripción	Elección del nivel de dificultad que se desea para jugar la partida.
Precondiciones	Usuario identificado como jugador creado previamente.
Post-condiciones	Se seleccionará el nivel de dificultad de la partida a jugar.
Escenario	1. Aparece la pantalla "seleccionar_opciones". 2. Se selecciona el botón de la dificultad con la que se desee

	jugar. 3. Se pulsa botón Empezar.
Condiciones de fallo	Que no se seleccione ningún nivel de dificultad

Tabla 59. CU - 10

CU-11	
Título	Seleccionar Comunidad Tanteo
Actores	usuario_identificado
Descripción	Elección de la posibilidad si se desea jugar con las reglas de tanteo de la comunidad de Madrid o de la comunidad de Extremadura.
Precondiciones	Usuario identificado como jugador creado previamente.
Post-condiciones	Se seleccionará las reglas de tanteo durante la partida a jugar.
Escenario	1. Aparece la pantalla "seleccionar_opciones". 2. Se selecciona la opción de Madrid o Extremadura según se desee. 3. Se pulsa botón Empezar.
Condiciones de fallo	Ninguna

Tabla 60. CU - 11

CU-12	
Título	Cambiar Nombres Participantes
Actores	usuario_identificado
Descripción	Se cambiará el nombre de los participantes de la partida.
Precondiciones	Usuario identificado como jugador creado previamente.
Post-condiciones	Los participantes de la partida tendrán el nombre seleccionado.
Escenario	1. Aparece la pantalla "opciones_partida". 2. Se pulsa botón Cambiar Nombres.

	3. Aparece pantalla “cambiar_nombres”. 4. Se modifican los nombres de los participantes que se desee. 5. Se pulsa botón Cambiar Nombres.
Condiciones de fallo	Se repitan los nombres de algunos participantes.

Tabla 61. CU - 12

CU-13	
Título	Jugar Partida
Actores	usuario_identificado
Descripción	Empezar a jugar una partida
Precondiciones	Usuario identificado como jugador creado previamente y haber seleccionado las opciones de la partida.
Post-condiciones	Aparecerá la pantalla “tapete_partida” y comenzará una partida.
Escenario	1. Aparece la pantalla “seleccionar_opciones”. 2. Se seleccionan las opciones de la partida. 3. Aparece pantalla “tapete_partida”
Condiciones de fallo	Ninguna

Tabla 62. CU - 13

CU-14	
Título	Cantar
Actores	usuario_identificado
Descripción	Jugador de la partida canta en algún palo.
Precondiciones	Usuario identificado como jugador creado previamente, opciones de la partida seleccionadas, que el jugador o su compañero ganen una baza y poseer rey y caballo del palo a cantar en las cartas del jugador.
Post-condiciones	El jugador realizará un cante.

Escenario	1. Aparece la pantalla “tapete_partida” 2. Jugador o compañero ganen una baza. 3. Pulsar botón Cantar y seleccionar el palo en el que se canta.
Condiciones de fallo	Que no se posea el caballo y el rey del palo a cantar.

Tabla 63. CU - 14

CU-15	
Título	Tirar Carta
Actores	usuario_identificado
Descripción	Tirar una carta de las que posee el jugador al tapete
Precondiciones	Estar jugando una baza de una partida y que sea el turno del jugador.
Post-condiciones	Se tirará la carta seleccionada por el jugador.
Escenario	1. Aparece la pantalla “tapete_partida”. 2. Toca turno del jugador en una baza. 3. Se selecciona carta a tirar 4. Se tira la carta que posee el jugador al tapete
Condiciones de fallo	Que no sea una de las posibles cartas que pueda tirar según las reglas del juego de la Cuatrola

Tabla 64. CU - 15

CU-16	
Título	Ordenar Cartas
Actores	usuario_identificado
Descripción	Ordenar las cartas que posee el jugador de la partida
Precondiciones	Estar jugando una partida y que el jugador posea cartas.
Post-condiciones	Se colocarán las cartas del jugador por palos y por puntuación.
Escenario	1. Aparece la pantalla “tapete_partida” 2. Se reparten cartas.

	3. Se pulsa botón Ordenar.
Condiciones de fallo	Ninguna

Tabla 65. CU - 16

CU-17	
Título	Ver Puntuación
Actores	usuario_identificado
Descripción	Se visualizará por pantalla la puntuación de la partida.
Precondiciones	Estar jugando una partida.
Post-condiciones	Aparecerá en pantalla la puntuación de la pareja del jugador y la de la pareja rival.
Escenario	1. Aparece la pantalla "tapete_partida". 2. Se muestran las puntuaciones de ambas parejas
Condiciones de fallo	Ninguna

Tabla 66. CU - 17

CU-18	
Título	Seleccionar Jugadas
Actores	usuario_identificado
Descripción	Seleccionar el tipo de jugada que quiere jugar el jugador con sus cartas
Precondiciones	Empezar a jugar una mano de una partida.
Post-condiciones	Jugar el tipo de jugada que seleccione el jugador.
Escenario	1. Aparece la pantalla "elegir_jugada". 2. Se pulsa botón de la jugada que quiera seleccionarse. 3. Se juega esa jugada.

Condiciones de fallo	Ninguna

Tabla 67. CU - 18

CU-19	
Título	Ver Puntuación
Actores	usuario_identificado
Descripción	Se visualizará por pantalla el pinte de la mano actual.
Precondiciones	Haber repartido las cartas de la mano de una partida.
Post-condiciones	Aparecerá en pantalla el pinte que haya salido en la mano al repartir.
Escenario	1. Aparece la pantalla "tapete_partida". 2. Se reparte una mano. 3. Se muestra el pinte que haya salido
Condiciones de fallo	Ninguna

Tabla 68. CU - 19

3.1.3. Requisitos Software

En este punto se realizará la definición de los requisitos software de modo que sea posible establecer, en todo momento, las condiciones que se deben cumplir para llegar a los objetivos que ha establecido el cliente con anterioridad. Estos, se establecerán de la siguiente forma:

YYY – XX			
Nombre			
Prioridad		**Necesidad**	
Verificabilidad		**Claridad**	
Estabilidad			
Descripción			
Relacionado con			

- **YYY:** Identificador que determinará si se trata de un requisito funcional o no funcional. Será RF si se trata de un requisito funcional y RNF si es un requisito no funcional.
- **XX:** Número identificativo del requisito.
- **Nombre:** Descriptor inicial del requisito.
- **Prioridad**. Medida que proporciona información de cara a realizar la planificación. Los valores pueden ser Alta, Media y Baja.
- **Necesidad**. Los valores pueden tomar los siguientes valores: Esencial, Deseable u Opcional.
- **Verificabilidad**. Es necesario que se pueda comprobar de forma fehaciente que los requisitos de usuario han sido implementados. Los valores pueden ser Alta, Media y Baja.
- **Claridad**. Se debe medir la ambigüedad de cada uno de los requisitos. Los valores pueden ser Alta, Media y Baja.
- **Estabilidad**. Se contemplará la permanencia a lo largo del proyecto.
- **Descripción:** Definición detallada para cada uno de los requisitos.

3.1.3.1. Requisitos funcionales

RF – 01			
Nombre	Mostrar Reglas		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	En la pantalla principal se implementará un botón que lleve a la pantalla donde se mostrarán las reglas		
Relacionado con	RUC - 01		

Tabla 69. RF - 01

RF – 02			
Nombre	Crear Jugador		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	En la pantalla crear_jugador se implementará un botón que comprobará al ser pulsado que el nombre introducido no está ya creado en la base de datos y si es así mostrará la pantalla seleccionar_opciones		
Relacionado con	RUC - 02, RUR - 02		

Tabla 70. RF - 02

RF – 03			
Nombre	Listar jugadores		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta

Estabilidad	Durante toda la vida del sistema
Descripción	Si hay usuarios creados en la aplicación se mostrarán el nombre de todos en la pantalla seleccionar_jugador en una lista desplegable para seleccionar el nombre del jugador con el que se quiere acceder o del que se quieren ver las estadísticas.
Relacionado con	RUC – 03, RUR - 03

Tabla 71. RF - 03

RF – 04			
Nombre	Seleccionar dificultad		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Si no se selecciona ningún nivel de dificultad cuando se presione el botón empezar, se mostrará un mensaje por pantalla avisando de que hay que seleccionar algún nivel de dificultad.		
Relacionado con	RUC – 06, RUR - 08		

Tabla 72. RF - 04

RF – 05			
Nombre	Comprobar Nombres Participantes		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Cuando se pulse el botón Cambiar Nombres en la pantalla cambiar_nombres se comprobará que todos los nombres de los participantes de la partida sean diferentes entre sí.		

Relacionado con	RUC – 08, RUR - 11

Tabla 73. RF - 05

RF – 06			
Nombre	Salir aplicación		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	En cualquier punto de la aplicación, un usuario podrá salir de la misma pulsando el botón de salir de su dispositivo móvil para salir de la pantalla en la que se encuentre.		
Relacionado con	RUC – 09, RUR - 13		

Tabla 74. RF - 06

RF – 07			
Nombre	Abandonar partida		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Un usuario podrá salir de una partida en juego desde el botón Abandonar de la pantalla opciones_partida.		
Relacionado con	RUC – 10, RUR - 14		

Tabla 75. RF - 07

RF – 08			
Nombre	Recargar Cantes		
Prioridad	Alta	**Necesidad**	Esencial

Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Al inicio de cada mano, se recargará la lista de cantes para actualizarlo según el pinte que haya sido repartido		
Relacionado con	RUC – 11, RUR - 15		

Tabla 76. RF - 08

RF – 09			
Nombre	Contabilizar Cantes		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Cuando se realice el conteo de puntos al final de cada mano, se sumará la puntuación de los cantes realizados por cada pareja.		
Relacionado con	RUC – 11		

Tabla 77. RF - 09

RF – 10			
Nombre	Verificar Carta		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Cuando un jugador selecciona una carta para tirar, la aplicación comprobará que es una carta aceptada por las reglas de la cuatrola.		
Relacionado con	RUC – 12, RUR - 18		

Tabla 78. RF - 10

RF – 11			
Nombre	Ordenar Cartas		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Las cartas de un jugador se ordenarán cuando el jugador lo desee.		
Relacionado con	RUC – 13, RUR - 19		

Tabla 79. RF - 11

RF – 12			
Nombre	Ver Puntuación		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Se mostrará por pantalla en todo momento la puntuación de cada partida. Esta puntuación será actualizada cada vez que se finalice una mano.		
Relacionado con	RUC – 14, RUR – 20		

Tabla 80. RF - 12

RF – 13			
Nombre	Seleccionar Jugadas		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		

Descripción	Un jugador decidirá si quiere jugarse una jugada especial o no.
Relacionado con	RUC – 15, RUR – 21, RUR - 22

Tabla 81. RF - 13

RF – 14			
Nombre	Elegir Jugadas		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	La aplicación elegirá la jugada que se juega cada uno de los participantes mediante un algoritmo de estudio según las cartas que posea dicho participante.		
Relacionado con	RUC – 15, RUR - 21		

Tabla 82. RF - 14

RF – 15			
Nombre	Representar Pinte Participantes		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Cuando un participante reparta, se mostrará la carta que le haya pintado visible como la última carta de las cartas de su mano hasta que éste la tire durante la mano.		
Relacionado con	RUC – 16, RUR – 24		

Tabla 83. RF – 15

RF – 16			
Nombre	Visualizar Pinte		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Un jugador tendrá constantemente visible la carta que sea el pinte en todo momento durante cada mano.		
Relacionado con	RUC – 16, RUR - 23		

Tabla 84. RF - 16

RF – 17			
Nombre	Visualizar estadísticas		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Un usuario puede elegir visualizar las estadísticas de cualquier jugador ya existente en la aplicación.		
Relacionado con	RUC – 17, RUR – 25, RUR - 26		

Tabla 85. RF - 17

3.1.3.2. Requisitos no funcionales

RNF – 01			
Nombre	Vista Reglas		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	En la pantalla de reglas se mostrarán las reglas de la Cuatrola y un botón para volver a la pantalla principal.		
Relacionado con	RUC - 01		

Tabla 86. RNF - 01

RNF – 02			
Nombre	Vista Crear Jugador		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	La pantalla crear_jugador constará de un cuadro de texto para introducir el nombre del jugador y de un botón para crearlo.		
Relacionado con	RUC - 02		

Tabla 87. RNF - 02

RNF – 03			
Nombre	Cuadros de texto de la aplicación		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta

Estabilidad	Durante toda la vida del sistema
Descripción	Todos los cuadros de texto de la aplicación tendrán una longitud máxima de 11 caracteres para introducir los nombres que serán alfanuméricos.
Relacionado con	RUC – 02, RUR – 01, RUR - 12

Tabla 88. RNF - 03

RNF – 04			
Nombre	Vista Seleccionar Jugador		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	En la pantalla seleccionar_jugador aparecerán los siguientes componentes: - Lista desplegable para visualizar el nombre de todos los jugadores. - Botón estadísticas que mostrará las estadísticas del usuario. - Botón de jugador nuevo que mostrará la pantalla crear_jugador. - Botón de Seleccionar jugador que comprobará si el jugador tiene partidas empezadas y no finalizadas para preguntar si quiere continuarla o mostrar la pantalla seleccionar_opciones.		
Relacionado con	RUC – 03, RUR - 04		

Tabla 89. RNF - 04

RNF – 05			
Nombre	Menú Reanudar Partida		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		

Descripción	Cuando un usuario se identifica, se le mostrará un menú preguntándole si quiere continuar la partida que tiene empezada. El menú dispone de dos opciones: - Si: mostrará la pantalla tapete_partida cargando los datos de la partida empezada y volviendo a repartir la última mano que se jugó. - No: se dará la partida empezada por perdida y aparecerá la pantalla seleccionar_opciones.
Relacionado con	RUC – 04, RUR – 05, RUR - 06

Tabla 90. RNF - 05

RNF – 06			
Nombre	RadioButton Comunidad Tanteo		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	En la pantalla seleccionar_opciones se utilizarán dos radioButton excluyentes entre sí para especificar si se quiere jugar con las reglas de Madrid o de Extremadura. Por defecto vendrá seleccionada la opción de Madrid		
Relacionado con	RUC – 05, RUR - 07		

Tabla 91. RNF - 06

RNF – 07			
Nombre	Botones dificultad		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Se mostrarán tres botones para seleccionar el nivel de dificultad llamados: Fácil, Medio y Difícil. La selección de éstos será excluyente		

	entre los tres. Por defecto no vendrá ninguno pulsado.
Relacionado con	RUC – 06, RUR - 09

Tabla 92. RNF - 07

RNF – 08			
Nombre	CheckBox Compañero Solo		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Se mostrará un checkBox para seleccionar la opción de que si quieres que tu compañero se vaya solo o no. Por defecto vendrá seleccionado.		
Relacionado con	RUC – 07, RUR - 10		

Tabla 93. RNF – 08

RNF – 09			
Nombre	Vista Selección Opciones		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	La pantalla seleccionar_opciones dispondrá de los siguientes componentes: - Un radioGroup: dentro de éste se encontrarán dos radioButton excluyentes para seleccionar la comunidad de tanteo. - Tres botones con los diferentes niveles de dificultad. - Un checkBox para seleccionar si el compañero puede irse solo. - Un botón llamado Empezar que mostrará la pantalla tapete_partida.		

Relacionado con	RUC – 05, RUC – 06, RUC – 07, RUR – 09, RUR - 10

Tabla 94. RNF - 09

RNF – 10			
Nombre	Vista Cambiar nombres		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	La pantalla cambiar_nombres dispondrá de los siguientes componentes: - Tres cuadros de texto: para introducir el nombre de cada uno de los tres participantes. - Botón Cambiar Nombres: mostrará la pantalla tapete_partida en la misma situación que estaba la partida actual.		
Relacionado con	RUC – 08		

Tabla 95. RNF - 10

RNF – 11			
Nombre	Menú Abandonar Partida		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Cuando un usuario pulsa el botón Abandonar de la pantalla opciones_partida, se le mostrará un menú preguntándole si quiere abandonar la partida que se encuentra jugando. El menú dispone de dos opciones: - Si: se dará la partida en juego por perdida y aparecerá la pantalla principal. - No: mostrará la pantalla tapete_partida en el estado en el que estaba la partida en juego.		

Relacionado con	RUC – 10, RUR - 14

Tabla 96. RNF - 11

RNF – 12			
Nombre	Vista Cantes		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Se mostrará en la pantalla tapete_partida una lista desplegable con todos los tipos de cantes posibles en cada mano		
Relacionado con	RUC – 11, RUR – 15		

Tabla 97. RNF - 12

RNF – 13			
Nombre	Habilitación Cantes		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Sólo podrá seleccionarse la lista desplegable de los cantes cuando haya finalizado una baza en la que la haya ganado el jugador o el compañero del jugador de la partida. Sólo se permitirá un cante por baza y participante.		
Relacionado con	RUC – 11, RUR – 16, RUR - 17		

Tabla 98. RNF - 13

RNF – 14			
Nombre	Tiempo Cantes		
Prioridad	Alta	**Necesidad**	Deseable
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Si el jugador tiene la posibilidad de realizar un cante, se aumentará el tiempo al recoger las cartas al final de una baza ganada por la pareja del jugador para que posea más tiempo para seleccionar el cante.		
Relacionado con	RUC – 11, RUR - 16		

Tabla 99. RNF - 14

RNF – 15			
Nombre	Mensaje Tirar Carta		
Prioridad	Alta	**Necesidad**	Deseable
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Si el jugador selecciona una carta para tirar no permitida en la baza se mostrará un mensaje por pantalla y no dejará tirar dicha carta.		
Relacionado con	RUC – 12, RUR - 18		

Tabla 100. RNF - 15

RNF – 16			
Nombre	Botón Ordenar Cartas		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		

Descripción	Se mostrará en la pantalla tapete_partida un botón Ordenar que ordenará las cartas cuando sea pulsado. Las cartas se ordenarán por palos en el siguiente orden: oros, copas, espadas y bastos. Dentro de cada palo, se ordenarán las cartas por puntuación.
Relacionado con	RUC – 13, RUR - 19

Tabla 101. RNF - 16

RNF – 17			
Nombre	Visualización Puntuación		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Se mostrará la puntuación de las parejas de la partida en dos cuadros de texto en la pantalla tapete_partida. Uno se colocará en la esquina inferior izquierda y llevará el tanteo de la pareja rival. Otro en la esquina inferior derecha y llevará el tanteo de la pareja del compañero.		
Relacionado con	RUC – 14, RUR - 20		

Tabla 102. RNF - 17

RNF – 18			
Nombre	Contenedor Pinte		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	El pinte se representará en la pantalla tapete_partida mediante un ImageView, siempre visible durante toda la mano, que contendrá la imagen de la carta que haya pintado durante cada mano.		
Relacionado con	RUC – 16, RUR - 23		

Tabla 103. RNF – 18

RNF – 19			
Nombre	Clasificación Estadísticas		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Las estadísticas de cada jugador serán mostradas por niveles de dificultad. Se realizarán las estadísticas para el nivel Fácil, para el nivel Medio y para el nivel Difícil por separado.		
Relacionado con	RUC – 17, RUR - 25		

Tabla 104. RNF - 19

RNF – 20			
Nombre	Campos Estadísticas		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Los campos sobre los que se realizarán las estadísticas son: - **Partidas Jugadas:** son el total de partidas jugadas. Dentro de este campo se dividirán por ganadas o perdidas y el porcentaje de partidas ganadas. - **Puntos Totales:** son el total de puntos jugados en todas las partidas. Dentro de este campo se dividirán por el número de ganados o perdidos y el porcentaje de puntos ganados. - **Solos:** son el total de Solos que ha echado el jugador en todas las partidas. Dentro de este campo se dividirán por el número de ganados o perdidos y el porcentaje de Solos ganados. - **Cuatrolas:** son el total de Cuatrolas que ha echado el jugador en todas las partidas. Dentro de este campo se dividirán por el número de ganadas o perdidas y el porcentaje de Cuatrolas ganadas.		

	- **Quintolas:** son el total de Quintolas que ha echado el jugador en todas las partidas. Dentro de este campo se dividirán por el número de ganadas o perdidas y el porcentaje de Quintolas ganadas.
Relacionado con	RUC – 17, RUR - 26

Tabla 105. RNF – 20

RNF – 21			
Nombre	Selección Jugadas		
Prioridad	Alta	**Necesidad**	Esencial
Verificabilidad	Alta	**Claridad**	Alta
Estabilidad	Durante toda la vida del sistema		
Descripción	Para seleccionar jugadas se mostrará la pantalla elegir_jugada. Esta pantalla dispondrá de un menú con cuatro opciones: - **Solo**: el jugador indicará su deseo de jugarse un solo. - **Cuatrola**: el jugador indicará su deseo de jugarse una Cuatrola. - **Quintola**: el jugador indicará su deseo de jugarse una Quintola. - **Nada**: el jugador indicará su deseo de no jugarse ninguna jugada especial.		
Relacionado con	RUC – 15, RUR – 21, RUR - 22		

Tabla 106. RNF – 21

3.1.4. Matriz de trazabilidad Requisitos Software – Requisitos de usuario

Una vez recogidos y analizados tanto los requisitos de usuario como los requisitos software, ha de comprobarse que todo requisito de usuario esté contemplado por al menos un requisito software.

Para hacer esta comprobación, se utilizan matrices de trazabilidad. Una matriz de trazabilidad tiene en su eje vertical el conjunto de identificadores de requisitos de usuario, tanto de capacidad como de restricción, y en su eje horizontal, los identificadores de requisitos software.

Si recordamos del apartado anterior, los requisitos software tienen un campo llamado Relacionado con que indica el requisito de usuario evaluado o implementado por el requisito software en cuestión. Siendo así, la matriz de trazabilidad marcará cada requisito de usuario contemplado por un requisito software concreto, siempre y cuando éste haya sido asignado en el campo Relacionado con correspondiente.

En la matriz, todos los requisitos de usuario quedan cubiertos por al menos un requisito software, lo que lleva a afirmar que todos los requisitos establecidos por el usuario serán implementados y, por tanto, no es necesario hacer un nuevo estudio de requisitos software.

	RUC-01	RUC-02	RUC-03	RUC-04	RUC-05	RUC-06	RUC-07	RUC-08	RUC-09	RUC-10	RUC-11	RUC-12	RUC-13	RUC-14	RUC-15	RUC-16	RUC-17
RF-01	X																
RF-02		X															
RF-03			X														
RF-04						X											
RF-05								X									
RF-06									X								
RF-07										X							
RF-08											X						
RF-09											X						
RF-10												X					
RF-11													X				
RF-12														X			
RF-13															X		
RF-14															X		
RF-15																X	
RF-16																X	
RF-17																	X
RNF-01	X																
RNF-02		X															

RNF-03		X															
RNF-04			X														
RNF-05				X													
RNF-06					X												
RNF-07						X											
RNF-08							X										
RNF-09					X	X	X										
RNF-10								X									
RNF-11										X							
RNF-12											X						
RNF-13											X						
RNF-14											X						
RNF-15												X					
RNF-16													X				
RNF-17														X			
RNF-18																X	
RNF-19																	X
RNF-20																	X
RNF-21															X		

Tabla 107. Matriz Trazabilidad Requisitos Software - Requisitos Capacidad

	RUR-01	RUR-02	RUR-03	RUR-04	RUR-05	RUR-06	RUR-07	RUR-08	RUR-09	RUR-10	RUR-11	RUR-12	RUR-13
RF-02		X											
RF-03			X										
RF-04								X					
RF-05											X		
RF-06													X
RNF-03	X											X	
RNF-04				X									
RNF-05					X	X							
RNF-06							X						
RNF-07									X				
RNF-08										X			
RNF-09									X	X			

Tabla 108. Matriz Trazabilidad Requisitos Software - Requisitos Restricción (I)

	RUR-14	RUR-15	RUR-16	RUR-17	RUR-18	RUR-19	RUR-20	RUR-21	RUR-22	RUR-23	RUR-24	RUR-25	RUR-26
RF-07	X												
RF-08		X											
RF-09													
RF-10					X								
RF-11						X							
RF-12							X						
RF-13								X	X				
RF-14								X					
RF-15											X		
RF-16										X			
RF-17												X	X
RNF-11	X												
RNF-12		X											
RNF-13			X	X									
RNF-14			X										
RNF-15					X								
RNF-16						X							
RNF-17							X						
RNF-18										X			
RNF-19												X	
RNF-20													X
RNF-21								X	X				

Tabla 109. Matriz Trazabilidad Requisitos Software - Requisitos Restricción (II)

3.2. Diseño

3.2.1. Diseño de la Arquitectura

La aplicación *Cuatrola Pro* cuenta con un reparto claro de funciones: una capa para la presentación (interfaz de usuario), otra para el cálculo (donde se encuentra el modelo de negocio) y otra para el acceso a datos.

En el desarrollo de software es imprescindible la utilización de patrones de diseño que otorguen a las aplicaciones características que les permitan ser fácilmente flexibles, evolutivas y de fácil mantenimiento.

Este proyecto se desarrolla utilizando el patrón Modelo-Vista-Controlador (MVC), el cual se adapta perfectamente a la arquitectura Cliente-Servidor y permite hacer una separación entre la parte gráfica y los procesos lógicos y de datos de las aplicaciones, de acuerdo a la arquitectura de tres niveles existente. En el patrón MVC existen tres elementos:

- **Modelo:** es el conjunto de clases que representan la información del mundo real que el sistema debe reflejar.
- **Vista:** es la encargada de la representación visual del modelo al usuario.
- **Controlador:** recibe, trata y responde a los eventos enviados por el usuario. Interactúa tanto con el modelo como con la vista.

En la *ilustración 12* se representa el flujo de control del patrón MVC.

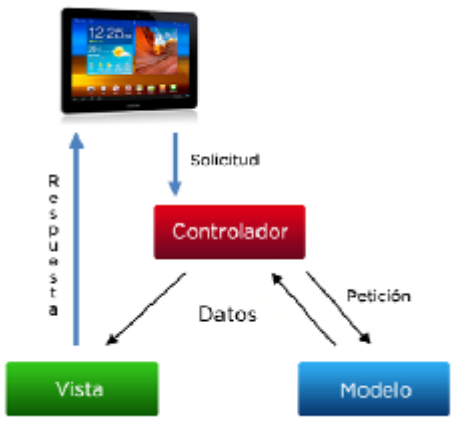

Ilustración 11. Flujo de control del modelo-vista-controlador

El Modelo-Vista-Controlador tiene una arquitectura que separa los datos, la interfaz de usuario y la lógica de control en capas diferentes, por lo que la evolución por separado de cada una de ellas es más fácil y aumenta la reutilización y la capacidad de flexibilidad.

Este patrón de diseño ofrece una gran transparencia al modificar el sistema ya que cada uno de los subsistemas incluye funcionalidades distintas de la aplicación y por lo tanto, serán independientes unos de otros, con lo cual, si falla uno, no tiene por qué fallar el resto.

En la *ilustración 13*, se representan los componentes de la aplicación.

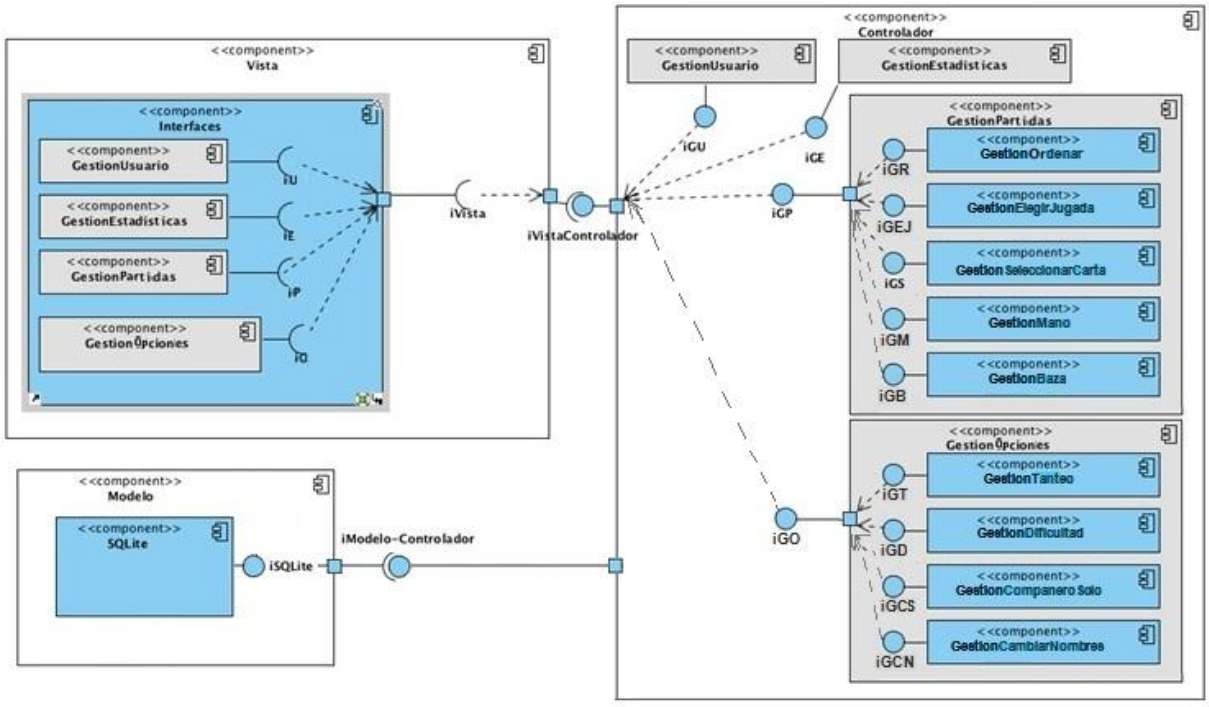

Ilustración 12. Diagrama de componentes

3.2.1.1. Modelo

Este componente contiene la representación específica a la información con la que el sistema va a tratar, de esta forma, la lógica de los datos garantiza la integridad de los mismos.

Podemos observar que este componente es el encargado de realizar las operaciones responsables de la gestión del almacenamiento y consulta de datos sobre la base de datos, así como insertar nuevos datos, modificarlos y borrarlos si fuera preciso.

SQLite será el encargado de recibir cada uno de los eventos por parte del controlador y acceder a la base de datos para realizar la operación que fue solicitada

3.2.1.2. Vista

El subsistema encargado de presentar el modelo de una forma que permita al usuario interactuar con la aplicación es la interfaz. Por lo tanto la vista incluye en gran medida el formato visual de la información, para lo cual deben usarse hojas de estilo.

Dentro de la ilustración podemos observar que el componente de la interfaz muestra las acciones que el usuario de Cuatrola Pro será capaz de realizar, divididos en subsistemas disponibles. Así mismo cada subsistema tiene un componente correspondiente en la capa del controlador.

Las interfaces son las siguientes:

- **GestionUsuario:** Mostrará todas las vistas relacionadas con el tratamiento de los usuarios. Su identificador será **iU**.
- **GestionEstadisticas:** Mostrará las vistas necesarias para representar las estadísticas de un usuario. Su identificador será **iE**.
- **GestionPartidas:** Mostrará todas las vistas necesarias para representar las partidas en juego. Su identificador será **iP**.
- **GestionOpciones:** Mostrará todas las vistas que se encarguen de seleccionar las opciones de una partida. Su identificador será **iO**.

3.2.1.3. Controlador

Y por último la capa del controlador se encarga de responder a todos los eventos o acciones que el usuario de *Cuatrola Pro* invoca en el modelo del sistema. Esta capa es capaz de procesar las entradas que el usuario realiza, estableciendo una comunicación entre el modelo y la vista. Es por esto que es la capa que guarda la lógica de cómo opera la aplicación.

- **GestionUsuario:** este componente gestiona cada una de las funcionalidades relacionadas con la información de los usuarios. Su identificador será **iGU**.
- **GestionEstadisticas:** este componente gestiona el cálculo de las estadísticas de cada usuario. Su identificador será **iGE**.
- **GestionPartidas:** Este componente es el encargado de gestionar todos los datos que haya que manipular durante una partida. Su identificador será **iGP**. Se subdivide en los siguientes componentes:
  - *GestionOrdenar*: Este componente se encargará de la gestión necesaria para ordenar las cartas de un jugador durante una partida. Su identificador será **iGOR**.
  - *GestionElegirJugada*: Este componente es el encargado de gestionar la elección por los participantes de una partida de si juegan una jugada especial o no. Su identificador será **iGEJ**.

- *GestionSeleccionarCarta*: Este componente se encargará de gestionar si una carta seleccionada para ser tirada está permitida y la tirará o mostrará un mensaje según corresponda. Su identificador será **iGS**.
- *GestionMano*: Este componente gestionará todos los cálculos internos e inserciones en la base de datos que se realicen en una mano. Su identificador será **iGM**.
- *GestionBaza*: Este componente gestionará todos los cálculos internos e inserciones en la base de datos que se realicen en una baza. Su identificador será **iGB**.

- **GestionOpciones:** Este componente se encarga de gestionar toda la información referente a la selección de las opciones en una partida. Su identificador será **iGO**. Se subdivide en los siguientes componentes:
  - *GestionTanteo*: Este componente gestiona la selección de la comunidad de la que seguirán las reglas del tanteo de las partidas. Su identificador será **iGT**.
  - *GestionDificultad*: Este componente gestiona la selección de la dificultad de una partida. Su identificador será **iGD**.
  - *GestionCompaneroSolo*: Este componente gestiona la selección de si el compañero del jugador de la partida se puede jugar jugadas especiales o no. Su identificador será **iGCS**.
  - *GestionCambiarNombres*: Este componente gestionará todas las verificaciones e inserciones en la base de datos necesarias para cambiar el nombre de los participantes de una partida. Su identificador será **iGCN**.

3.2.1.4. Matriz trazabilidad componentes / requisitos de usuario

La siguiente matriz de trazabilidad verifica que todos los requisitos han sido recogidos por los componentes, ya que si algún requisito quedara sin ser asignado a al menos un componente, significaría que el sistema no ha tenido en cuenta dicho requisito y habría de solucionarlo de inmediato.

	iU	iE	iP	iO	iGU	iGE	iGP	iGOR	iGEJ	iGS	iGM	iGB	iGO	iGT	iGD	iGCS	iGCN
RUC-01	X																
RUC-02	X				X												
RUC-03	X				X												
RUC -04	X				X												
RUC -05				X									X	X			
RUC -06				X									X		X		
RUC-07				X									X			X	
RUC-08				X									X				X
RUC-09					X		X										
RUC-10							X										
RUC-11			X				X				X	X					
RUC-12			X				X			X		X					
RUC-13			X				X	X				X					

RUC-14			X				X					X					
RUC-15			X				X		X								
RUC-16			X														
RUC-17		X				X											
RUR-01	X			X													
RUR-02					X												
RUR-03	X				X												
RUR-04	X																
RUR-05							X				X						
RUR-06							X				X						
RUR-07				X													
RUR-08				X									X		X		
RUR-09				X													
RUR-10				X													
RUR-11													X				X
RUR-12	X			X													
RUR-13							X										
RUR-14			X														
RUR-15			X														
RUR-16			X				X				X	X					
RUR-17							X					X					
RUR-18			X				X			X							
RUR-19			X				X	X									
RUR-20			X														
RUR-21			X														
RUR-22							X		X								
RUR-23			X														
RUR-24			X														
RUR-25		X				X											
RUR-26		X				X											

Tabla 110. Matriz trazabilidad componentes / requisitos de usuario

3.2.2. Diseño de clases

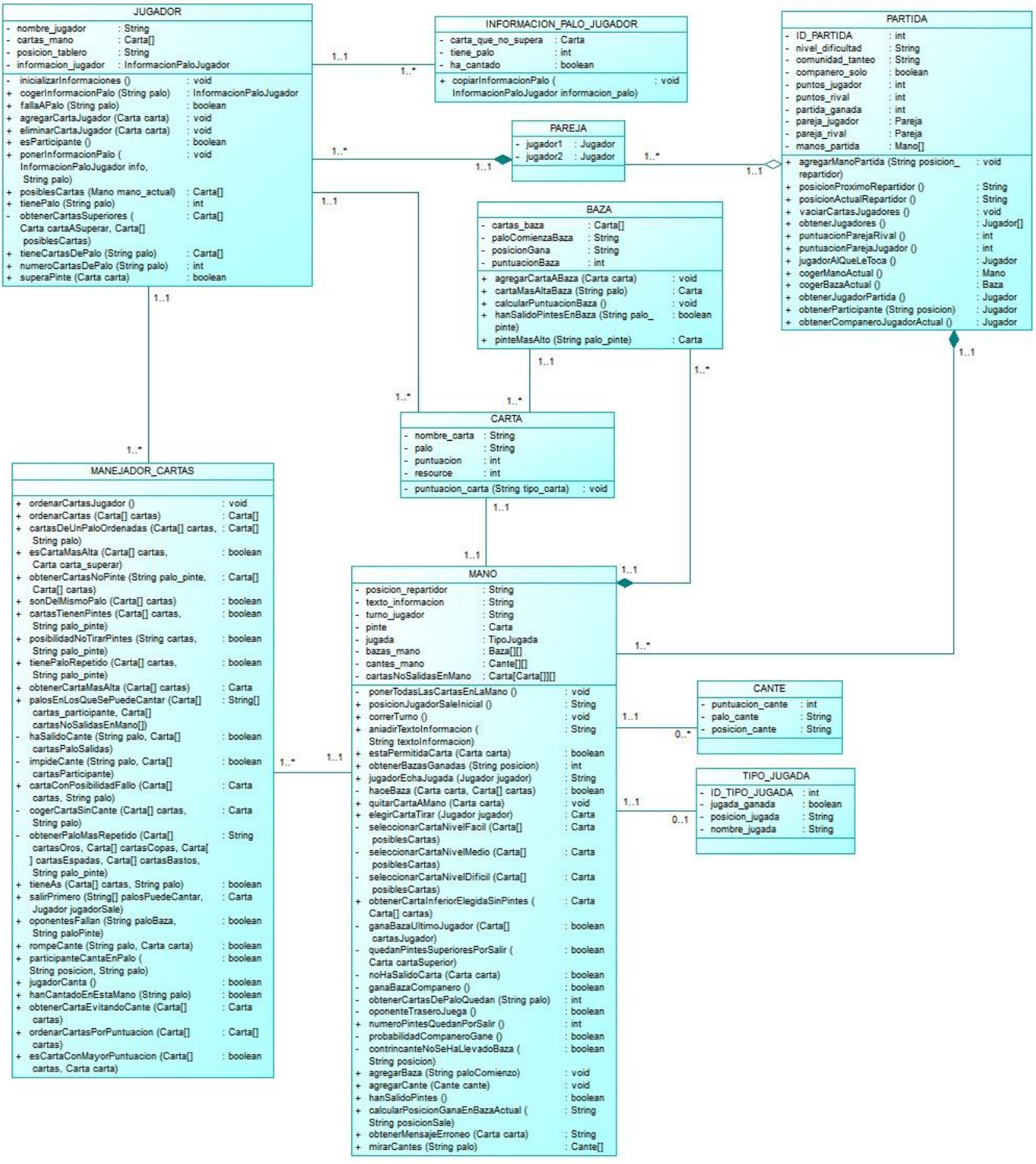

Ilustración 13. Diagrama de Clases

A continuación se mostrará una tabla con cada una de las clases explicando sus atributos y operaciones, así como una breve descripción de la funcionalidad que desempeñará en la aplicación.

Identificador: **CL - 01**	
Nombre de la clase	***JUGADOR***
DESCRIPCIÓN CLASE	
Clase encargada de gestionar a los jugadores de una partida	
ATRIBUTOS	
Nombre_jugador <String> Nombre del jugador actual de la partida	
Cartas_mano <Array de Carta> Array con las cartas que posee el jugador en su mano	
Posicion_tablero <String> Posición que posee el jugador en el tablero de la partida.	
Informacion_jugador <Array de InformacionPaloJugador> Array que posee la información conocida del jugador con respecto a las cartas que haya tirado durante la mano. Por cada posición del Array se guardará la información de un palo.	
MÉTODOS	
inicializarInformaciones () Método que se encarga de inicializar el Array que contiene las Informaciones de los palos del jugador.	
cogerInformacionPalo(String palo) Devuelve la Información del palo del jugador del palo que se haya solicitado	
fallaAPalo (String palo) Comprueba si por la información del palo del jugador se sabe si va a fallar al palo introducido como argumento.	
agregarCartaJugador (Carta carta) Agrega la carta indicada a la mano del jugador.	
eliminarCartaJugador (Carta carta) Elimina la carta indicada de la mano del jugador.	
esParticipante() Devuelve true si el jugador no es el jugador actual de la partida y false si lo es.	
ponerInformacionPalo (InformacionPaloJugador info, String palo) Introduce la información del palo de un jugador en el palo introducido	
posiblesCartas (Mano mano_actual) Devuelve un Array con todas las posibles cartas que el jugador podría tirar en la mano actual.	
tienePalo (String palo) Devuelve mediante un entero a otro jugador si por la información de la partida se sabe si el jugador tiene un palo. Devolverá 0 cuando no se sepa, 1 si sí tiene el palo, 2 si no lo tiene y 3 si puede tenerlo.	
obtenerCartasSuperiores (Carta cartaASuperar, Carta[] posiblesCartas) Devuelve un Array de cartas con todas las cartas de las posibles superan a la carta a superar.	
tieneCartasDePalo(String palo) Este método recorre las cartas de la mano del jugador e indica si tiene cartas de ese palo.	
numeroCartasDePalo (String palo) Devuelve el número de cartas del palo solicitado que tiene el jugador en la mano	

superaPinte (Carta carta)
Devuelve si el jugador supera al pinte que ha salido en la mano con alguna de sus cartas.

Tabla 111. CL - 01: JUGADOR

Identificador: **CL - 02**	
Nombre de la clase	***INFORMACION_PALO_JUGADOR***
DESCRIPCIÓN CLASE	
Clase que guarda la información de un palo de un jugador según vaya avanzando la partida	
ATRIBUTOS	
Carta_que_no_supera <Carta> Carta a la que un jugador no ha superado en ese palo	
Tiene_palo <int> Indica si el jugador tiene ese palo. Puede ser: 0, no se sabe si lo tiene; 1, sí que lo tiene; 2, no lo tiene; y 3, puede que lo tenga.	
Ha_cantado <boolean> Si el jugador ha cantado en ese palo	
MÉTODOS	
copiarInformacionPalo (InformacionPaloJugador InformacionPalo) Copia los valores del palo de información que le llega.	

Tabla 112. CL - 02: INFORMACION_PALO_JUGADOR

Identificador: **CL - 03**	
Nombre de la clase	***PAREJA***
DESCRIPCIÓN CLASE	
Clase que guarda los jugadores de una pareja	
ATRIBUTOS	
Jugador1 <Jugador> Primer jugador de la pareja	
Jugador2 <Jugador> Segundo jugador de la pareja	

Tabla 113. CL - 03: PAREJA

Identificador: **CL - 04**	
Nombre de la clase	***PARTIDA***
DESCRIPCIÓN CLASE	
Clase que guarda la información de una partida	
ATRIBUTOS	
ID_PARTIDA <int> Identificador de la partida	
Nivel_dificultad <String> Nivel de dificultad de la partida	
Comunidad_tanteo <String> Comunidad de la que se siguen las reglas de tanteo de la partida	
Companero_solo <boolean> Indicador de si el compañero del jugador puede jugarse una jugada especial	

Puntos_jugador <int> Puntos que posee la pareja del jugador de la partida
Puntos_rival <int> Puntos que posee la pareja rival de la partida
Partida_ganada <int> Indicador de si la partida está ganada, perdida o en curso
Pareja_jugador <Pareja> Identificador de la pareja del jugador
Pareja_rival <Pareja> Identificador de la pareja rival
Manos_partida <Array de Mano> Array con todas las manos de la partida
MÉTODOS
agregarManoPartida (String posicion_repartidor) Agrega una mano nueva al array de manos.
posicionProximoRepartidor() Devuelve la posición del repartidor de la siguiente mano
posicionActualRepartidor () Devuelve la posición del repartidor de la mano actual
vaciarCartasJugadores() Elimina todas las cartas de las manos de los jugadores
obtenerJugadores() Devuelve un array con todos los jugadores de la partida
puntuacionParejaRival() Devuelve la puntuación de la pareja rival
puntuacionParejaJugador() Devuelve la puntuación de la pareja del jugador
jugadorAlQueLeToca() Devuelve el jugador que posee el turno
cogerManoActual () Devuelve la mano actual que se está jugando
cogerBazaActual () Devuelve la baza actual que se está jugando
obtenerJugadorPartida () Devuelve el jugador que está jugando la partida
obtenerParticipante (String posicion) Devuelve el jugador que se encuentra en la posición solicitada
obtenerCompaneroJugadorActual () Devuelve el jugador que es el compañero del jugador que posea el turno actual

Tabla 114. CL - 04: PARTIDA

Identificador: **CL - 05**	
Nombre de la clase	***BAZA***
DESCRIPCIÓN CLASE	
Clase que guarda la información de las bazas de la mano de una partida	
ATRIBUTOS	
paloComienzaBaza <String> Palo por el que comienza la baza	
Cartas_baza <Array de Carta> Array con las cartas que han salido en la baza	
posicionGana <String> Posición del jugador que gana la baza	
puntuacionBaza <int> Puntuación de todas las cartas que posee la baza	
MÉTODOS	
agregarCartaABaza (Carta carta) Agrega la carta recibida a la baza	
cartaMasAltaBaza (String palo) Devuelve la carta más alta que se encuentra en la baza del palo solicitado	
calcularPuntuacionBaza () Calcula la puntuación de la baza	
hanSalidoPintesEnBaza (String palo_pinte) Devuelve si han salido cartas del palo solicitado o no.	
pinteMasAlto (String palo_pinte) Devuelve la carta más alta que tenga la baza del palo que se ha solicitado	

Tabla 115. CL - 05: BAZA

Identificador: **CL - 06**	
Nombre de la clase	***CARTA***
DESCRIPCIÓN CLASE	
Clase que guarda la información de las cartas de la partida	
ATRIBUTOS	
Nombre_carta <String> Nombre de la carta.	
palo <String> Palo de la carta.	
puntuacion <int> Puntuación de la carta.	
resource <int> Entero del identificador del recurso que contiene la imagen de esa carta en la aplicación.	
MÉTODOS	
puntuacion_carta (String tipo_carta) Devuelve la puntuación de una carta según su tipo. El tipo puede ser: AS, TRES, REY, CABALLO O SOTA.	

Tabla 116. CL - 06: CARTA

Identificador: **CL - 07**	
Nombre de la clase	***MANEJADOR_CARTAS***
DESCRIPCIÓN CLASE	
Clase de apoyo para realizar operaciones sobre las cartas	
MÉTODOS	
ordenarCartasJugador () Realiza la función de ordenar las cartas del jugador de la partida.	
ordenarCartas(Carta[] cartas) Devuelve el array de cartas que le llega ordenado.	
cartasDeUnPaloOrdenadas (Carta[] cartas, String palo) Devuelve un array con las cartas del palo solicitado ordenadas del array de cartas que le llega.	
esCartaMasAlta (Carta[] cartas, Carta carta_superar) Devuelve si la carta que le llega es la más alta del array de cartas que le llega.	
obtenerCartasNoPinte (String palo_pinte, Carta[] cartas) Devuelve un array con todas las cartas, del array de cartas que le llega, que no sean del palo que le llega.	
sonDelMismoPalo (Carta[] cartas) Devuelve si las cartas del array que le llegan son todas del mismo palo	
cartasTienenPintes (Carta[] cartas, String palo_pinte) Indica si las cartas que le llegan en el array alguna posee el palo del pinte solicitado.	
posibilidadNoTirarPintes (Carta[] cartas, String palo_pinte) Indica si entre las posibles cartas que le llegan, hay alguna que no sea del palo solicitado.	
tienePaloRepetido (Carta[] cartas, String palo_pinte) Indica si hay más de una carta en el array que le llega del palo solicitado	
palosEnLosQueSePuedeCantar (Carta[] cartas_participante, Carta[] cartasNoSalidasEnMano[]) Comprueba entre las cartas que han salido en la mano y las cartas del jugador actual los palos en los que se puede realizar un cante.	
haSalidoCante(String palo, Carta[] cartasPaloSalidas) Indica si ha salido el caballo o el rey del palo solicitado en las cartas que ya se han tirado.	
impideCante (String palo, Carta[] cartasParticipante) Comprueba si un participante posee entre sus cartas el caballo o el rey del palo solicitado.	
cartaConPosibilidadFallo (Carta[] cartas, String posicion_participante) Devuelve una carta que pueda hacer que el resto de los jugadores fallen.	
cogerCartaSinCante (Carta[] cartas, String palo) Coge una carta de las cartas posibles del palo solicitado que no rompa un cante del jugador	
obtenerPaloMasRepetido (Carta[] cartasOros, Carta[] cartasCopas, Carta[] cartasEspadas, Carta[] cartasBastos, String palo_pinte) Devuelve el palo más repetido de las cartas del jugador que no rompa ningún cante	
tieneAs(Carta[] cartas, String palo) Devuelve si las cartas que le llegan tiene el as del palo solicitado	
salirPrimero (String[] palosPuedeCantar, Jugador jugadorSale) Devuelve la carta que tirará el primer jugador de una baza	
oponentesFallan (String paloBaza, String paloPinte) Comprueba si los oponentes no tienen cartas de la baza y tienen cartas del palo del pinte para saber si fallarán en la información de los palos conocida de sus contrincantes.	

rompeCante (String posicion_jugador, Carta carta) Devuelve si un jugador canta en el palo de la carta y la carta solicitada le rompería el cantar.
participanteCantaEnPalo (String posicion, String palo) Devuelve si el jugador situado en esa posición canta en el palo solicitado
jugadorCanta () Devuelve si el jugador de la partida tiene algún cante.
hanCantadoEnEstaMano (String palo) Devuelve si ya se ha cantado en el palo solicitado durante la mano.
obtenerCartaEvitandoCantes (Carta[] cartas) Devuelve una carta de las posibles cartas que no sea ni un caballo ni un rey.
ordenarCartasPorPuntuacion (Carta[] cartas) Devuelve un array de cartas ordenadas por la puntuación de éstas de menor a mayor
esCartaConMayorPuntuacion (Carta[] cartas, Carta carta) Devuelve si es la carta más grande del array de cartas solicitado

Tabla 117. CL - 07: MANEJADOR_CARTAS

Identificador: **CL - 08**	
Nombre de la clase	***MANO***
DESCRIPCIÓN CLASE	
Clase que guarda la información de las manos de una partida	
ATRIBUTOS	
Posicion_repartidor <String> Posición del repartidor de la mano	
Texto_informacion <String> Texto de información de la mano	
Turno_jugador <String> Turno del jugador de la mano	
pinte <Carta> Carta que ha pintado en la mano	
Jugada <TipoJugada> Tipo de jugada que se ha jugado en la mano	
Bazas_mano <Array de Baza> Array con todas las bazas de la mano	
Cantes_mano <Array de cante> Array con todos los cantes de la mano	
cartasNoSalidasEnMano <Array de Arrays de Carta> Array con las cartas que no hay salido en la mano por palos.	
MÉTODOS	
ponerTodasLasCartasEnLaMano () Inicializa todas las cartasNoSalidasEnMano guardando los valores de todas las cartas	
posicionJugadorSaleInicial () Devuelve la posición del jugador que tiene el primer turno	
correrTurno () Cambia el turno al siguiente jugador que le toque	
aniadirTextoInformacion (String textoInformacion)	

Agrega texto al texto de información
estaPermitidaCarta (Carta carta)
Comprueba si está permitida la carta seleccionada por el jugador de la partida
obtenerBazasGanadas (String posicion)
Devuelve un array con las bazas que ha ganado el jugador de esa posición
jugadorEchaJugada (Jugador jugador)
Devuelve el tipo de jugada que se juega un participante de la partida
haceBaza (Carta carta, Carta[] cartas)
Comprueba si la carta solicitada es la más alta de las que quedan por salir del palo de esa carta.
quitarCartaAMano (Carta carta)
Quita la carta solicitada de las cartas que faltan por salir en la mano
elegirCartaTirar (Jugador jugador)
Devuelve la carta que tirará un participante de la partida.
seleccionarCartaNivelFacil (Carta[] posiblesCartas)
Devuelve la carta que tirará un participante de la partida con el algoritmo de selección del nivel fácil
seleccionarCartaNivelMedio (Carta[] posiblesCartas)
Devuelve la carta que tirará un participante de la partida con el algoritmo de selección del nivel medio.
seleccionarCartaNivelDifícil (Carta[] posiblesCartas)
Devuelve la carta que tirará un participante de la partida con el algoritmo de selección del nivel difícil.
obtenerCartaInferiorElegidaSinPintes (Carta[] cartas)
Devuelve la carta inferior que no haga baza y que no rompa un cante de entre las cartas que le llegan.
ganaBazaUltimoJugador (Carta[] cartasJugador)
Devuelve si el último participante que tira en la baza se lleva la baza.
quedanPintesSuperioresPorSalir (Carta cartaSuperior)
Devuelve si quedan pintes más altos que no se hayan tirados que la carta solicitada.
noHaSalidoCarta (Carta carta)
Devuelve si la carta ya ha sido tirada en esta mano o no.
ganaBazaCompanero ()
Devuelve si gana la baza el compañero del jugador con el turno actual.
obtenerCartasDePaloQuedan (String palo)
Devuelve un array con las cartas del palo solicitado que quedan por salir
oponenteTraseroJuega()
Cuando es el turno del segundo tirador y hay jugada especial, devuelve si el compañero de atrás juega o no.
numeroPintesQuedanPorSalir()
Devuelve el número de pintes que aún no han sido tirados en la mano.
probabilidadCompaneroGane ()
Devuelve si es probable que el compañero del jugador que posee el turno actual pueda ganar la baza o no.
contrincanteNoSeHaLlevadoBaza (String posicion)
Calcula si un contrincante del jugador que posee el turno actual se ha llevado alguna baza.
agregarBaza (String paloComienzo)

Agrega una baza a la mano.
agregarCante (Cante cante) Agrega un cante a la mano
hanSalidoPintes () Devuelve si han salido pintes en la mano
calcularPosicionGanaEnBazaActual (String posicionSale) Calcula la posición del ganador de la baza actual
obtenerMensajeErroneo (Carta carta) Devuelve el texto de por qué no se puede tirar la carta seleccionada
mirarCantes (String palo) Devuelve un array de Cantes con todos los cantes que se han realizado en la mano

Tabla 118. CL - 08: MANO

Identificador: **CL - 09**	
Nombre de la clase	***CANTE***
DESCRIPCIÓN CLASE	
Clase para guardar la información de un cante	
ATRIBUTOS	
Puntuacion_cante <int> Puntuación del cante	
Palo_cante <String> Palo del cante	
Posicion_cante <String> Posicición del jugador que ha realizado el cante	

Tabla 119. CL - 09: CANTE

Identificador: **CL - 10**	
Nombre de la clase	***TIPO_JUGADA***
DESCRIPCIÓN CLASE	
Clase para guardar la información del tipo de una jugada	
ATRIBUTOS	
ID_TIPO_JUGADA <int> Identificador del tipo de jugada	
Jugada_ganada <Boolean> Indica si la jugada se ha perdido o ganado. 0 será perdida y 1 será ganada.	
Posicion_jugada <String> Posición del jugador que se ha jugado la jugada especial	
Nombre_jugada <String> Nombre de la jugada especial que se ha jugado.	

Tabla 120. CL - 10: TIPO_JUGADA

3.2.3. Diseño del modelo

Se ha realizado un estudio exhaustivo de los requisitos, casos de uso y diagrama de clases de la aplicación y con toda la información recogida se ha confeccionado el modelo entidad – relación y el modelo relacional para confeccionar la disposición de la base de datos para poder almacenar toda la información necesaria para la aplicación.

3.2.3.1. Modelo Entidad/Relación – Relación Datos

El diagrama del modelo de entidad – relación ha sido diseñado en la aplicación yEd Graph Editor. Se muestra en la siguiente ilustración.

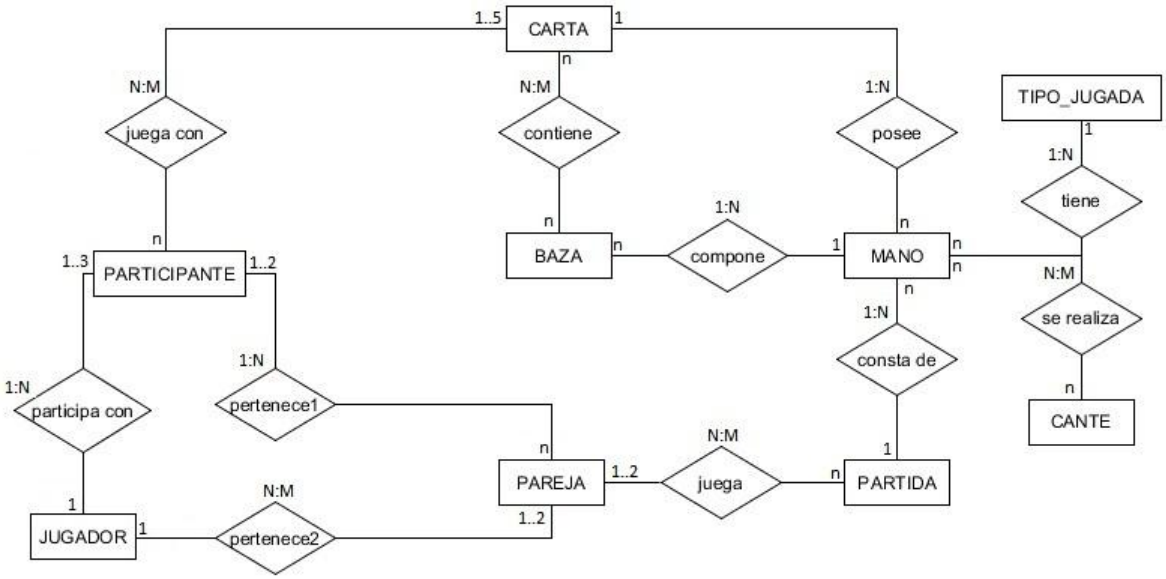

Ilustración 14. Modelo Entidad - Relación

3.2.3.2. Modelo Relacional de datos

Para realizar el modelo relacional de Datos se ha utilizado el programa PowerDesigner. A continuación se mostrará la ilustración de dicho modelo y una tabla por cada entidad con una breve descripción de su funcionalidad y de qué atributos consta.

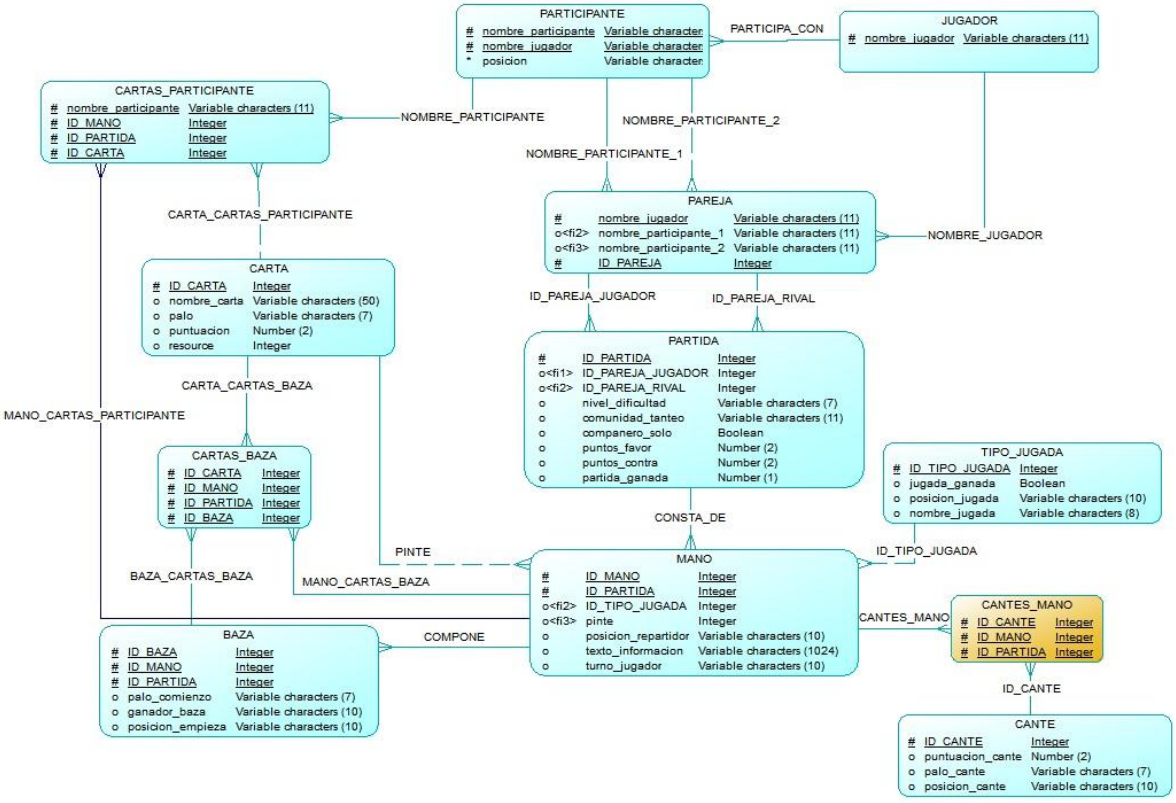

Ilustración 15. Modelo Relacional

Nombre entidad	JUGADOR
Descripción	Entidad que guardará los nombres de todos los jugadores que se den de alta en la aplicación
Atributos	*nombre_jugador:* Nombre del jugador.

Tabla 121. Entidad JUGADOR

Nombre entidad	PARTICIPANTE
Descripción	Entidad que guardará los nombres de todos los participantes de la aplicación
Atributos	*nombre_jugador:* nombre del jugador al que pertenece el participante.
	nombre_participante: nombre del participante.
	posicion: posición que ocupa el participante en el tablero.

Tabla 122. Entidad PARTICIPANTE

Nombre entidad	PAREJA
Descripción	Entidad que guardará las parejas de la aplicación
Atributos	*nombre_jugador:* nombre del jugador al que pertenece la pareja.
	nombre_participante_1: nombre del primer participante de la pareja.
	nombre_participante_2: nombre del segundo participante de la pareja.
	ID_PAREJA: Identificador de la pareja.

Tabla 123. Entidad PAREJA

Nombre entidad	CARTA
Descripción	Entidad que guardará la información de las cartas de la aplicación
Atributos	*ID_CARTA:* Identificador de la carta.
	nombre_carta: nombre de la carta.
	palo: palo de la carta.
	puntuacion: puntuación de la carta.
	resource: El identificador del recurso que recibe la carta en la aplicación.

Tabla 124. Entidad CARTA

Nombre entidad	BAZA
Descripción	Entidad que guardará la información de cada una de las bazas de una mano.
Atributos	*ID_BAZA:* Identificador de la baza.
	ID_MANO: Identificador de la mano.
	ID_PARTIDA: Identificador de la partida.
	palo_comienzo: palo de comienzo de la baza.
	ganador_baza: posición del ganador de la baza.
	posicion_empieza: posición del jugador que empieza la baza.

Tabla 125. Entidad BAZA

Nombre entidad	TIPO_JUGADA
Descripción	Entidad que guardará la información de las jugadas especial que se juegan en una mano.
Atributos	*ID_TIPO_JUGADA:* Identificador del tipo de jugada.
	jugada_ganada: Booleano que indica si la jugada se ha ganado o no. Cero será perdida y uno ganada.
	posicion_jugada: posición del jugador que juega la jugada especial.
	nombre_jugada: nombre de la jugada especial. Podrá ser SOLO, CUATROLA ó QUINTOLA.

Tabla 126. Entidad TIPO_JUGADA

Nombre entidad	CANTE
Descripción	Entidad que guardará la información de los cantes que se realicen en las manos de las partidas.
Atributos	*ID_CANTE:* Identificador del cante.
	puntuacion_cante: puntuación del cante realizado. Será 20 ó 40.
	palo_cante: palo en el que se realiza el cante.
	posicion_cante: posición del participante que realiza el cante.

Tabla 127. Entidad CANTE

Nombre entidad	MANO
Descripción	Entidad que guardará la información de cada una de las manos de una partida.
Atributos	*ID_MANO:* Identificador de la mano.
	ID_PARTIDA: Identificador de la partida.
	ID_TIPO_JUGADA: Identificador de la jugada especial de esta mano. Será nulo si en esa mano no se ha realizado ninguna jugada especial.
	pinte: Identificador de la carta que haya pintado en esa mano.
	posicion_repartidor: Posición del repartidor de la mano.
	texto_informacion: texto de la información de la mano.
	turno_jugador: posición del jugador que tiene el turno en esa mano.

Tabla 128. Entidad MANO

Nombre entidad	PARTIDA
Descripción	Entidad que guardará la información de cada partida.
Atributos	*ID_PARTIDA:* Identificador de la partida.
	ID_PAREJA_JUGADOR: Identificador de la pareja del jugador que juega la partida.
	ID_PAREJA_RIVAL: Identificador de la pareja rival del jugador que juega la partida.
	nivel_dificultad: indica el nivel de dificultad de la partida. Los valores son fácil, medio y difícil.
	comunidad_tanteo: indica la comunidad de la que se seguirán las reglas de tanteo durante la partida. Los valores posibles son Madrid o Extremadura.
	companero_solo: Booleano que indica si el compañero del jugador de la partida puede jugarse una jugada especial. Será cero cuando no pueda y uno si puede jugarla.
	puntos_favor: Puntuación de la pareja del jugador de la partida.
	puntos_contra: puntuación de la pareja rival del jugador de la partida.
	partida_ganada: Entero que indica el estado de la partida. Será cero cuando la pareja del jugador de la partida haya perdido, uno cuando haya ganado y dos cuando no haya acabado.

Tabla 129. Entidad PARTIDA

Nombre entidad	CARTAS_PARTICIPANTE
Descripción	Entidad que guardará la información de las cartas que posee cada participante durante una partida inacabada.
Atributos	*nombre_participante:* nombre del participante que posee las cartas.
	ID_MANO: Identificador de la mano.
	ID_PARTIDA: Identificador de la partida.
	ID_CARTA: Identificador de la carta.

Tabla 130. Entidad CARTAS_PARTICIPANTE

Nombre entidad	CARTAS_BAZA
Descripción	Entidad que guardará la información de las cartas que han salido en una baza durante una partida inacabada.
Atributos	*ID_BAZA:* Identificador de la baza.

	ID_MANO: Identificador de la mano.
	ID_PARTIDA: Identificador de la partida.
	ID_CARTA: Identificador de la carta.

Tabla 131. Entidad CARTAS_BAZA

Nombre entidad	CANTES_MANO
Descripción	Entidad que guardará la información de los cantes que se han realizado en una mano durante una partida.
Atributos	*ID_CANTE:* Identificador del cante.
	ID_MANO: Identificador de la mano.
	ID_PARTIDA: Identificador de la partida.

Tabla 132. Entidad CANTES_MANO

3.2.3.3. Especificación de Necesidades de Migración de Datos y Carga Inicial

No existen necesidades de migración de datos puesto que se trata de una aplicación nueva que no posee datos antiguos.

La carga inicial se encargará de realizar las inserciones en la base de datos de los datos estáticos de la aplicación. Estos datos se corresponden con la inserción de todas las cartas existentes en la tabla CARTA según el modelo de datos presentado.

3.2.4. Diseño de interfaz

En este apartado se detallan las diferentes pantallas por las que puede navegar un usuario en *Cuatrola Pro*.

- **Pantalla principal**.

Ilustración 16. Pantalla principal

Cuando el usuario accede a la aplicación, se muestra la siguiente pantalla. La pantalla *principal* consta de dos botones:

- Botón **Jugar**: Cuando es pulsado, muestra la pantalla *crear_jugador* si no hay ningún jugador dado de alta en la aplicación o la pantalla *seleccionar_jugador* si ya hay algún usuario creado.
- Botón **Reglas**: al ser pulsado aparece la pantalla *reglas*.

- **Pantalla reglas**.

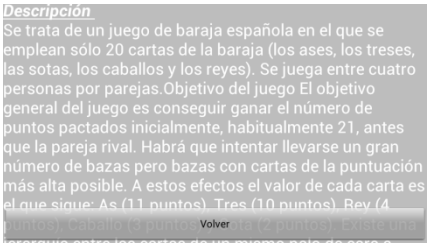

Ilustración 17. Pantalla reglas

La pantalla *reglas* muestra las reglas de la aplicación y posee un botón:

- Botón **Volver**: Cuando es pulsado, muestra la pantalla *principal* de nuevo.

- **Pantalla crear_jugador**.

Ilustración 18. Pantalla crear_jugador

La pantalla *crear_jugador* es la pantalla destinada a crear los jugadores en la aplicación. Posee un cuadro de texto y un botón:

- Cuadro de texto: En él se escribirá el nombre del jugador y tendrá una longitud máxima de 11 caracteres que serán alfanuméricos.
- Botón **Crear Jugador**: Al ser pulsado comprobará que el nombre introducido en el cuadro de texto anterior contenga de 1 a 11 caracteres y que no se encuentre ya dado de alta en la base de datos. Si se cumplen estas dos comprobaciones, se crea el usuario con ese nombre y se accede a la pantalla *seleccionar_opciones*.

- **Pantalla seleccionar_jugador**.

Ilustración 19. Pantalla seleccionar_jugador

La pantalla *seleccionar_jugador* es la pantalla destinada a identificar a los jugadores existentes en la aplicación. Consta de:

- Lista desplegable: En ella se cargarán todos los jugadores dados de alta en la aplicación y se podrá seleccionar al que se desee.
- Botón **Estadísticas**: Al ser pulsado se mostrará la pantalla *estadísticas* con las estadísticas del jugador que se encuentre seleccionado en la lista desplegable.

- Botón **Jugador Nuevo**: Al pulsarse se mostrará la pantalla *crear_jugador*.
- Botón **Seleccionar Jugador**: Al ser pulsado comprobará si el nombre seleccionado en la lista desplegable anterior posee partidas inacabadas. Si las tiene, muestra el menú de la pantalla *reanudar_partida* y si no, accede a la pantalla *seleccionar_opciones*.

- **Pantalla seleccionar_opciones**.

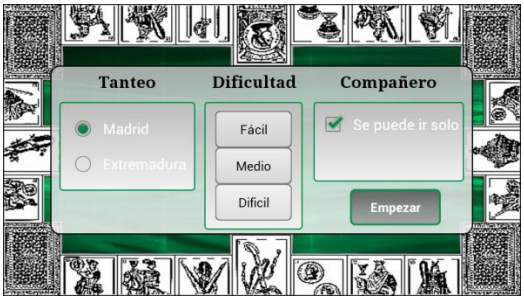

Ilustración 20. Pantalla seleccionar_opciones

La pantalla *seleccionar_opciones* es la pantalla destinada a seleccionar las opciones de una partida. Consta de:

- Dos radioButtons excluyentes: Se seleccionará el que se desee según si se quiere jugar con las opciones de la comunidad de Madrid o de Extremadura.
- Tres botones **Nivel de Dificultad**: Según el nivel que se desee, se pulsará Fácil, Medio o Difícil. Si se pulsa uno, todos los demás serán deseleccionados inmediatamente.
- CheckBox **Se puede ir solo**: Si aparece pulsado significará que el compañero puede jugarse jugadas especiales. Si no, el compañero no se jugará ninguna jugada especial.
- Botón **Empezar**: Al ser pulsado comprobará que se ha seleccionado algún nivel de dificultad y si es así creará una nueva partida con estas opciones y mostrará la pantalla *tapete_partida*. Si no hay ningún nivel de dificultad seleccionado, mostrará un mensaje por pantalla avisando de que hay que seleccionar algún nivel.

- **Pantalla tapete_partida**.

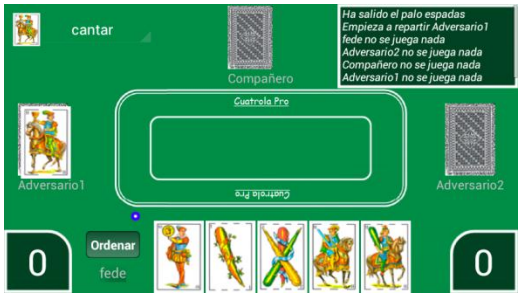

Ilustración 21. Pantalla tapete_partida

La pantalla *tapete_partida* es la pantalla destinada a jugar partida. Consta de:

- Un ImageView: Se encuentra situado arriba a la izquierda y mostrará durante toda una mano el pinte que se haya repartido.
- Una lista desplegable **cantar**: en esta lista desplegable se seleccionará en qué palo se desea cantar cuando el jugador o su compañero ganen una baza.
- Dos cuadros de texto: Situados en las esquinas inferiores de la pantalla y mostrarán las puntuaciones de las parejas de la partida. El cuadro inferior izquierdo mostrará la puntuación de la pareja rival. El cuadro inferior derecho mostrará la puntuación de la pareja del jugador.
- Botón **Ordenar**: Al ser pulsado ordenará las cartas de la mano del jugador.
- Un cuadro de texto en la esquina superior derecha: este cuadro de texto mostrará la información relevante de la partida.
- Cinco ImageView: cada uno de ellos poseerá cada una de las cartas del jugador. Cuando el jugador tenga su turno dentro de una baza, tendrá que seleccionar el que contenga la carta que desea seleccionar para ser tirada.
- Las cartas de los participantes de la partida: se verán las cartas de los participantes de la partida y el pinte que le haya salido al jugador que reparta.
- Cuatro cuadros de texto: contendrán los nombres de todos los participantes de la partida.

- **Pantalla opciones_partida**.

Ilustración 22. Pantalla opciones_partida

La pantalla *opciones_partida* es la pantalla destinada a seleccionar las posibles opciones dentro de una partida en juego. Consta de:

- Un botón **Cambiar Nombres**: Al ser pulsado se mostrará la pantalla *cambiar_nombres*.
- Un botón **Continuar Partida**: Al pulsarse se regresará a la pantalla *tapete_partida*.
- Un botón **Abandonar Partida**: Al ser pulsado muestra el menú de la pantalla *abandonar_partida*.

- **Pantalla cambiar_nombres**.

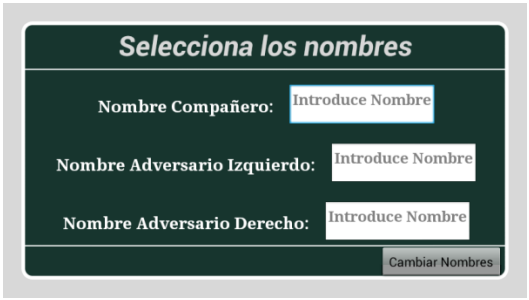

Ilustración 23. Pantalla cambiar_nombres

La pantalla *cambiar_nombres* es la pantalla destinada a modificar el nombre de los participantes existentes en la partida. Consta de:

- Tres cuadros de texto: En ellos se escribirá el nombre de los participantes que se desee cambiar y tendrán una longitud máxima de 11 caracteres que serán alfanuméricos.
- Un botón **Cambiar Nombres**: Al pulsarse se comprobará que tras el cambio realizado, los cuatro nombres de los participantes de la partida son diferentes entre sí. Si son diferentes se realizará el cambio de nombres y volverá a la pantalla *tapete_partida*. Si no mostrará un mensaje diciendo que hay nombres repetidos.

- **Pantalla abandonar_partida**.

Ilustración 24. Pantalla abandonar_partida

La pantalla *abandonar_partida* es la pantalla destinada a decidir si se desea abandonar una partida en juego, dando ésta por perdida. Consta de:

- Un botón **Si**: Si se pulsa se indicará que la partida actual se ha perdido y se regresará a la pantalla *principal*.
- Un botón **No**: Si se pulsa se regresará a la pantalla *tapete_partida* y se continuará la partida en juego.

- **Pantalla reanudar_partida**.

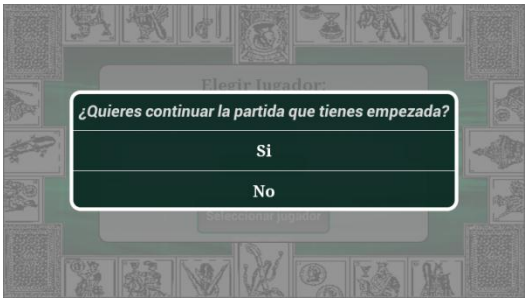

Ilustración 25. Pantalla reanudar_partida

La pantalla *reanudar_partida* es la pantalla destinada a decidir si se desea continuar una partida que se había iniciado anteriormente. Consta de:

- Un botón **Si**: Si se pulsa se indicará que se desea continuar la partida y cargará los datos de la partida inacabada en la pantalla *tapete_partida* y la mostrará por pantalla.
- Un botón **No**: Si se pulsa se dará la partida iniciada anteriormente cómo perdida y se mostrará la pantalla *seleccionar_opciones*.

- **Pantalla estadisticas**.

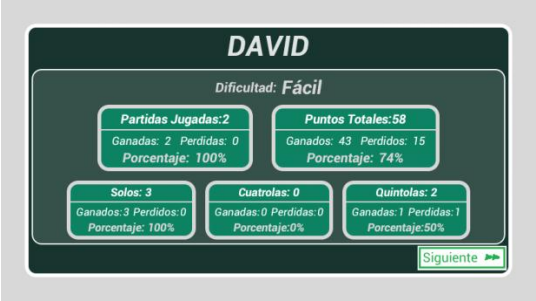

Ilustración 26. Pantalla estadisticas

La pantalla *estadisticas* es la pantalla destinada a mostrar las estadísticas de un jugador en el juego por niveles de dificultad. Consta de:

- Cuadros de etiquetas: estas etiquetas se recargan con los datos de las partidas que se encuentran en la base de datos del jugador seleccionado.
- Botones de movimiento: según en el nivel en el que se encuentre dispondrá de botones para avanzar de nivel, retroceder o volver. Al pulsar el botón volver del nivel de dificultad Difícil, se regresará a la pantalla *seleccionar_jugador*.

- **Pantalla elegir_jugada**.

Ilustración 27. Pantalla elegir_jugada

La pantalla *elegir_jugada* es la pantalla destinada a que el jugador seleccione el tipo de jugada que desea jugar. Consta de:

- Un botón de **opciones**: este botón se encuentra situado arriba de la pantalla al lado de las cartas del compañero. Al ser pulsado, mostrará la pantalla *opciones_partida*.
- Cuatro botones de **Seleccion Jugada**: según la jugada que desee el jugador seleccionará la opción que desee. Las opciones permitidas son: Solo, Cuatrola, Quintola y nada. Al pulsar alguna de ellas se volverá a la pantalla *tapete_partida* y se continuará jugando la mano en juego.

3.2.4.1. Diagrama de navegabilidad

En este apartado se muestran las relaciones entre las interfaces presentadas anteriormente. Para lograr una aplicación dinámica se ha llevado a cabo un sistema funcional y navegable. Los círculos representan en los botones que se pulsan y la línea unida a dicho círculo la pantalla que aparece. El esquema de comportamiento entre las pantallas se conforma de la siguiente manera:

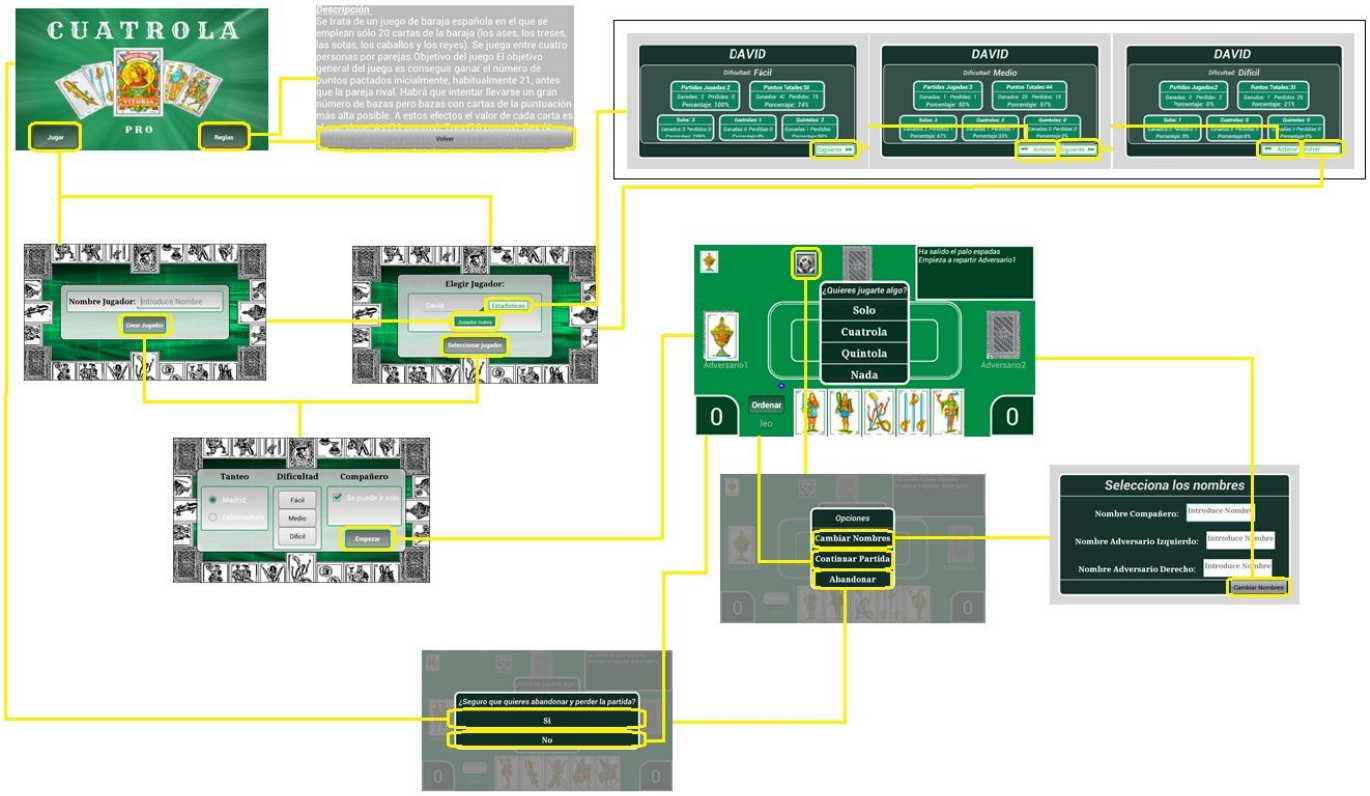

Ilustración 28. Diagrama de navegabilidad

3.3. Implementación

En la parte de implementación se explicará el modelo de proceso que se ha seguido a la hora de la implementación. También nos centraremos en explicar los algoritmos que han sido utilizados para la implementación de las partes más conflictivas de la aplicación.

3.3.1. Modelo de proceso

El ciclo de vida de un proyecto software se define como un marco de referencia en el que se incluyen los procesos, actividades y tareas que forman parte del proceso de desarrollo, mantenimiento y despliegue de un producto software [8].

El modelo de ciclo de vida es el modelo que contempla el estado de las fases que componen un proyecto software, indicando el orden en el que se realiza cada una de las tareas y los criterios de transición entre ellas.

El modelo de proceso utilizado en el presente proyecto es un modelo de proceso iterativo e incremental [9]. Este modelo se basa en una serie de fases realimentadas y aplicadas repetidamente con una filosofía iterativa [10]. Las tareas de desarrollo se organizan separadamente en iteraciones, con el fin de que en cada una de ellas se logre una pieza de software construida sobre la versión lograda en iteraciones previas. El funcionamiento de un ciclo iterativo incremental se muestra en la siguiente ilustración.

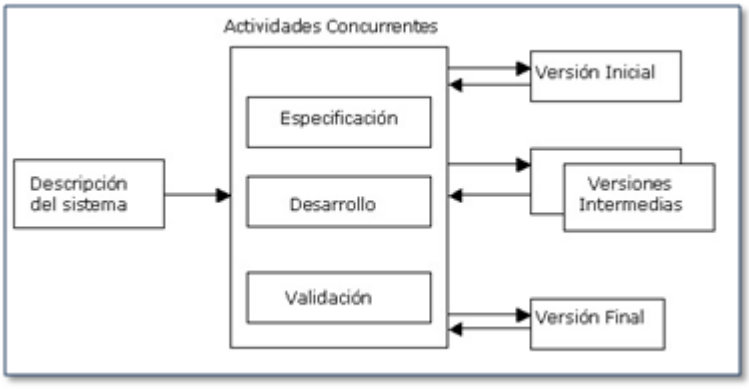

Ilustración 29. Esquema del funcionamiento de un ciclo iterativo incremental

Entre las ventajas del modelo de proceso iterativo e incremental se encuentran las siguientes:

- En cada iteración es posible obtener un producto parcial pero completamente operacional en lugar de una versión en la que se han reajustado los requisitos.
- Se reduce la tasa de fallo del proyecto.
- Se mejora la productividad del equipo de trabajo a través de la experiencia obtenida iteración tras iteración.

- Permite manejar la complejidad del proyecto, apuntando a la resolución de los problemas por partes.

El motivo por el que el presente proyecto pertenece al modelo de proceso iterativo e incremental es porque se han realizado varios incrementos hasta llegar a completar una iteración. Igualmente, se han realizado varias iteraciones hasta obtener el producto final.

En concreto, en el proceso de análisis se han realizado dos incrementos. Tras realizar el prototipado de la aplicación móvil en la fase de diseño, surgieron nuevos requisitos, que fueron añadidos al catálogo original. Tras agregar estos requisitos, se realizó un nuevo incremento en la fase de diseño, creando un nuevo prototipo que cumpliese con los nuevos requisitos. De igual manera, se realizó un segundo incremento de la fase de análisis al añadir nuevos requisitos tras realizar el segundo prototipo. No obstante, el catálogo de requisitos presentado en este documento se corresponde con la versión final del mismo.

3.3.2. Algoritmo de selección de jugada

A la hora de implementar el algoritmo encargado de seleccionar si un participante de la partida tiene cartas para jugarse una jugada especial se ha utilizado un algoritmo de poda basado en los casos que pueden sucederse según las cartas que posea el participante, la carta que haya pintado, el turno en el que el participante tenga que tirar carta en la primera baza de la mano, la posibilidad de que puedan cantarle en algún palo los contrincantes y si al compañero le ha pintado un pinte para saber que se jugaría con un pinte menos la mano. Dentro de las cartas de la mano del participante se comprobará el número de pintes que se tiene, el número de ases y el número de palos repetidos que posea.

Se ha generado un algoritmo con todas las posibilidades de éstas variantes según un orden de prioridad. La variante de más prioridad se ha considerado el número de pintes de la mano del participante, que puede oscilar entre uno y cinco. Para los casos extremos (uno y cinco pintes) no se ha considerado ninguna correlación con el resto de variables. Para los otros tres casos se han ido correlacionando las posibilidades de las otras variables para tomar la toma de la decisión de la jugada del participante, generándose así el método de poda según las condiciones de la partida.

3.3.3. Algoritmo de selección de carta

A la hora de decidir la carta a seleccionar se estudia el número de cartas posibles que puede tirar cada participante cuando posee el turno. Si la opción es una única carta, se tirará ésta sin tomar ninguna decisión. Si por el contrario hay más de una posibilidad, la decisión de la carta que se tirará dependerá del nivel de dificultad que posea la partida. Se explicará cada decisión en los siguientes subapartados.

3.3.3.1. Nivel fácil

En este nivel los participantes contrincantes del jugador de la partida tomarán la decisión de soltar la carta según el algoritmo de selección del nivel fácil y el compañero del jugador tomará sus decisiones según el algoritmo de selección del nivel difícil.

En este nivel se usa el algoritmo de “el peor caso posible”. Este algoritmo se implementa de manera que, de las cartas permitidas que puede tirar el participante, siempre tirará la carta de mayor puntuación posible. Si el participante entre sus opciones pudiese tirar una carta pinte o una que no sea pinte, siempre se elegirá la carta de mayor puntuación posible de todas aquellas que no sean del palo que haya pintado.

3.3.3.2. Nivel medio

En este nivel, tanto los participantes contrincantes del jugador de la partida como el participante compañero del jugador tomarán sus decisiones según el algoritmo de selección del nivel medio.

Se usa el algoritmo de aleatoriedad. Este algoritmo se implementa de manera que, de las cartas permitidas que puede tirar el participante, tirará una carta al azar, pudiendo resultar ésta la mejor o la peor opción posible. Si el participante entre sus opciones pudiese tirar una carta pinte o una que no sea pinte, siempre se elegirá una carta al azar de todas aquellas que no sean del palo que haya pintado.

3.3.3.3. Nivel difícil

En este nivel todos los participantes tomarán la decisión de soltar la carta según el algoritmo de selección del nivel difícil.

En este nivel se usa un algoritmo de poda basado en las condiciones que se pueden dar en una partida dependiendo de un número de variables y de la correlación entre éstas. Para el estudio de las variantes de este algoritmo se dispone de la información que un participante sabe de una partida mediante las cartas que hayan tirado anteriormente el resto de jugadores, si alguno ha cantado en algún palo, si los contrincantes pueden cantar y si la baza la va a ganar la pareja del participante al que le toca seleccionar la carta o la pareja contraria para decidir si es preferible tirar una carta que puntúe más que otra.

Inicialmente se observa la posición del participante que tiene que seleccionar la carta. Según ésta posición, se van analizando una serie de circunstancias.

- Si se trata del primer concursante de la mano, analizará las variables de en qué palos pueden cantar sus contrincantes, en qué palos puede cantar su compañero, las cartas que hagan baza de su mano, el número de cartas repetidas que posea de un palo y la probabilidad que tienen los participantes de fallar a algún palo para quitarles pintes.
- Si se trata del segundo jugador de la mano, analizará las variables de si la probabilidad de ganar la baza es de la carta que ha salido en la mano, si es probable que su compañero falle y tenga pintes para fallar, si es probable que falle el contrincante que aún no ha tirado la carta, si se está jugando jugada especial para saber que uno de los dos participantes que tiene por detrás no juega y de si puede tirar una carta que no le rompan ningún cante si lo tuviese.

- Si se trata del tercer jugador de la mano, analizará las variables de si se está jugando una jugada especial para saber si es él el último en tirar en la baza, si es probable que la baza la gane su compañero, si el participante de detrás puede fallar y si tiene la opción de que no le rompan un cante.
- Si se trata del último jugador de la mano, analizará las variables de si la baza va a favor de su compañero, si él gana la baza con sus opciones, si las cartas que tiene para tirar podrían ganar una baza posterior y el número de pintes que quedan en la mano.

Se analizarán todas éstas variantes, según el caso en el que nos encontremos, otorgándoles un orden de prioridad a cada variante o la correlación entre ellas que se decide mediante un razonamiento basado en casos, por lo que se ha decidido la prioridad de cada una de las variantes o de la correlación entre ellas a partir del estudio de lo que ha sucedido en casos similares en otras ocasiones y cuáles de estas variantes eran más determinantes a la hora de seleccionar una carta u otra.

3.3.4. Algoritmo de repartición de cartas

El algoritmo de reparto de cartas se realiza de forma aleatoria. Se cuenta con un array de veinte posiciones en las que habrá una carta diferente en cada posición encontrándose todas las cartas que conforman una mano.

Se empieza a repartir por turnos y a la hora de entregar una carta a cada participante, se obtiene un número aleatorio entre todas las posiciones de las que conste el array. Una vez elegida una carta, se agrega a la mano del participante y se elimina del array de todas las cartas que quedan por repartir. En el siguiente turno se obtendrá otro número aleatorio entre número de posiciones de que conste el array y así sucesivamente hasta que el array que poseía las veinte cartas iniciales se vacíe.

3.4. Pruebas

Una vez desarrollada la aplicación se ha de probar que funcione correctamente. Para ello se desarrollan tests que aseguren el correcto funcionamiento del sistema comprobando que cada módulo funcione como es esperado, así como ciertos tests que permitan comprobar bajo qué condiciones el sistema tarda en presentar respuestas.

3.4.1. Pruebas de funcionalidad

Estas pruebas se desarrollan con el objetivo de verificar la funcionalidad de los distintos elementos que componen el sistema. Se realizan en un entorno controlado y comprobando que la respuesta de las operaciones coincidía con los requisitos establecidos en la fase de análisis.

Fueron repetidas a lo largo del desarrollo del proyecto y corrigiéndose los distintos errores encontrados cada vez, hasta obtener los resultados esperados.

Por longitud, sólo se detallan algunas de las pruebas realizadas. Para facilitar la lectura, serán recogidas en tablas cuyo formato será:

PF_XX		
Nombre:		
Descripción:		
Acciones:		
Precondiciones	***Resultado Esperado***	***Resultado Final***

Tabla 133. Plantilla de pruebas funcionales

- **ID**: Identificador de la prueba. Su nomenclatura ha de seguir el formato PF_XX, donde XX será sustituido por el número de la prueba.
- **Nombre**: nombre de la prueba.
- **Descripción**: descripción del objetivo de la prueba.
- **Acciones**: pasos a seguir para la realización de la prueba.
- **Resultados Esperados**: En este apartado se definen según las precondiciones que se den a la hora de realizar la prueba, los resultados que se esperan para que la prueba esté correcta y el resultado final obtenido.
    - **Precondición**: será la precondición que se pueda dar al iniciar la prueba.
    - **Resultado Esperado**: resultado que se desea.
    - **Resultado Final**: especifica si tras la prueba se ha obtenido el resultado esperado. Puede tomar los valores de Válido e Inválido.
    -

PF_01		
Nombre:	Comprobar redirección botones	
Descripción:	Se comprobará que al pulsarse los botones de navegabilidad de la aplicación redirigen correctamente a las pantallas conforme al diagrama de navegabilidad del punto 3.2.3.1.	
Acciones:	1. Pulsar un botón que sea de redirección 2. Comprobar que muestra la pantalla correspondiente	
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que exista en pantalla un botón de redirección	Muestra la pantalla correspondiente	Válido

Tabla 134. PF_01

PF_02		
Nombre:	Comprobar cambiar nombres participantes	
Descripción:	Se comprobará que al cambiar los nombres de los participantes no se puedan insertar más de once caracteres y que se cambian correctamente.	
Acciones:	1. Introducir el nombre de los participantes a cambiar en la pantalla *cambiar_nombres*. 2. Pulsar sobre botón **Cambiar Nombres**	
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que se intenten introducir más de 11 caracteres para los nombres	No dejará introducir ningún carácter después del onceavo carácter.	Válido
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que no se haya introducido ningún nombre	Volverá a la pantalla *partida_tapete* sin realizar ningún cambio.	Válido
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que haya algunos nombres de los participantes de la partida repetidos	Mostrará un mensaje por pantalla de error y no dejará cambiar los nombres.	Válido
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que todos los nombres de los participantes de la partida sean distintos	Cambia correctamente los nombres en la base de datos y actualiza los nombres de la partida en juego.	Válido

Tabla 135. PF_02

PF_03	
Nombre:	Comprobar se introducen opciones partida
Descripción:	Se comprobará que antes de comenzar la partida, el jugador haya elegido todas las opciones de ésta.
Acciones:	1. Elegir las opciones de la partida en la pantalla *seleccionar_opciones*. 2. Pulsar sobre botón **Empezar**

Precondiciones	***Resultado Esperado***	***Resultado Final***
Que no se haya seleccionado un nivel de dificultad	Mostrará un mensaje por pantalla de aviso de que tiene que seleccionar una dificultad y no dejará comenzar la partida.	Válido
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que se haya seleccionado un nivel de dificultad	Creará una partida nueva y dirigirá a la pantalla *partida_tapete*.	Válido

Tabla 136. PF_03

PF_04		
Nombre:	Comprobar crear jugador	
Descripción:	Se comprobará que al crear un jugador no se puedan insertar más de once caracteres y que se cree correctamente.	
Acciones:	1. Introducir el nombre del jugador que se desea crear en la pantalla *crear_jugador*. 2. Pulsar sobre botón **Crear Jugador**	
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que se intenten introducir más de 11 caracteres para el nombre	No dejará introducir ningún carácter después del onceavo carácter.	Válido
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que no se haya introducido ningún nombre	Mostrará aviso por pantalla de que tiene que introducir el nombre del jugador a crear sin hacer nada más.	Válido
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que haya ya un jugador con ese nombre creado en la aplicación	Mostrará un mensaje por pantalla de error de jugador ya creado y no dejará crear al jugador.	Válido
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que el nombre del jugador no esté creado aún	Creará al jugador insertando ese nombre en la base de datos y redirigirá a la pantalla *seleccionar_opciones*.	Válido

Tabla 137. PF_04

PF_05		
Nombre:	Comprobar listado jugadores	
Descripción:	Se comprobará si se carga correctamente la lista desplegable en la pantalla *seleccionar_jugador*.	
Acciones:	1. Mostrar pantalla *seleccionar_jugador* 2. Leer nombres de los jugadores existentes en la base de datos. 3. Cargar los nombres en la lista desplegable.	
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que exista algún jugador ya creado en la aplicación.	Muestra en la lista el nombre de todos los jugadores creados en la aplicación	Válido

Tabla 138. PF_05

PF_06		
Nombre:	Comprobar reanudar partida	
Descripción:	Se comprobará que al seleccionar un jugador para jugar tenga una partida empezada e inacabada.	
Acciones:	1. Seleccionar el nombre del jugador en la lista desplegable de la pantalla *seleccionar_jugador*. 2. Pulsar sobre botón **Seleccionar Jugador**	
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que el jugador tenga una partida empezada e inacabada	Se redirigirá a la pantalla *reanudar_partida*.	Válido
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que el jugador no tenga una partida empezada e inacabada	Se redirigirá a la pantalla *seleccionar_opciones*.	Válido

Tabla 139. PF_06

PF_07		
Nombre:	Comprobar abandonar partida	
Descripción:	Se comprobará que ocurre al salir de una partida.	
Acciones:	1. Estar jugando una partida. 2. Abandonar partida	
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que el jugador abandone la partida pulsando el botón **Si** de la pantalla *abandonar_partida*	Se modificará la partida empezada para ponerla como perdida y se redirigirá a la pantalla *principal*.	Válido
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que el jugador abandone la partida pulsando la opción salir con los botones del dispositivo	Se abandonará la partida sin darse ni por perdida ni por ganada y se redirigirá a la pantalla *principal*.	Válido

Tabla 140. PF_07

PF_08		
Nombre:	Comprobar cantes	
Descripción:	Se comprobará que al estar jugando la mano de una partida se realicen únicamente los cantes del jugador cuando esté jugando, su pareja gane una baza y tenga dicho cante en las cartas de la mano.	
Acciones:	1. Jugar partida, jugar la mano, poseer un cante en las cartas de la mano y que tu pareja o tú os llevéis la baza. 2. Pulsar sobre botón **Cantar** 3. Seleccionar el cante que se posee en las cartas de la mano	
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que el jugador no esté jugando la mano o que su pareja no acabe de ganar la baza actual.	El botón cantar aparecerá deshabilitado.	Válido
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que el jugador no posea el cante en las cartas de la mano.	Se mostrará mensaje de error y no se permitirá realizar el cante	Válido
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que el jugador posea el cante en las cartas de la mano, esté jugando la mano y su pareja acabe de ganar la baza actual.	Se mostrará mensaje de cante realizado y se contabilizará en el cálculo de puntos del final de la mano.	Válido

Tabla 141. PF_08

PF_09		
Nombre:	Comprobar posible carta seleccionada	
Descripción:	Se comprobará si un jugador ha seleccionado una carta que está permitida tirar por las reglas de la Cuatrola.	
Acciones:	1. Estar jugando una partida. 2. Tener el turno en una baza. 3. Seleccionar una carta a tirar.	
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que el jugador no haya seleccionado una carta que se ajusta a las reglas de la Cuatrola en su turno.	Mostrará mensaje del tipo de carta que tiene que tirar y no dejará que la carta seleccionada se tire.	Válido
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que el jugador haya seleccionado una carta que se ajusta a las reglas de la Cuatrola en su turno.	Tirará la carta al centro del tapete.	Válido

Tabla 142. PF_09

PF_10		
Nombre:	Comprobar ordenar cartas	
Descripción:	Se comprobará si un jugador puede ordenar las cartas de su mano.	
Acciones:	1. Estar jugando una partida. 2. Pulsar botón **Ordenar**.	
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que el jugador no tenga cartas en la mano	El botón Ordenar estará deshabilitado y no podrá ser pulsado	Válido
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que el jugador tenga cartas en la mano	Ordenará las cartas de la mano del jugador.	Válido

Tabla 143. PF_10

PF_11		
Nombre:	Comprobar cálculos partida	
Descripción:	Se comprobará que todos los cálculos de los puntos de una partida son correctos.	
Acciones:	1. Estar jugando una partida. 2. Acabar una mano.	
Precondiciones	***Resultado Esperado***	***Resultado Final***
Haber acabado de jugar una mano	Se actualizará la puntuación de la partida de la pareja que haya ganado con los puntos ganados según las reglas de la comunidad del tanteo	Válido

Tabla 144. PF_11

PF_12		
Nombre:	Comprobar seleccionar jugada especial	
Descripción:	Se comprobará que al seleccionar una jugada especial se modifiquen las condiciones de la partida.	
Acciones:	1. Estar jugando una partida. 2. Seleccionar una jugada especial en la pantalla *elegir_jugada*	
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que el participante seleccione jugar una jugada especial.	En esa mano no jugará el compañero del participante que ha seleccionado la jugada especial.	Válido

Tabla 145. PF_12

PF_13		
Nombre:	Comprobar visualizar pinte	
Descripción:	Se comprobará se visualiza la carta que haya pintado en la partida.	
Acciones:	1. Estar jugando una partida. 2. Acabar de repartir las cartas de una partida	
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que haya repartido un participante de la partida que no sea el jugador	Se mostrará la carta pinte arriba a la izquierda de la pantalla *tapete_partida* y detrás de las cartas de la mano del participante que ha repartido	Válido
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que haya repartido el jugador de la partida	Se mostrará la carta pinte arriba a la izquierda de la pantalla *tapete_partida*	Válido

Tabla 146. PF_13

PF_14		
Nombre:	Comprobar cálculo estadísticas	
Descripción:	Se comprobará se visualiza el cálculo de las estadísticas.	
Acciones:	1. Seleccionar un jugador en la pantalla *seleccionar_jugador*. 2. Pulsar botón **Estadísticas**.	
Precondiciones	***Resultado Esperado***	***Resultado Final***
Que haya algún jugador creado en la aplicación	Se mostrará la pantalla estadísticas con las *estadísticas* del jugador seleccionado.	Válido

Tabla 147. PF_14

3.4.2. Matriz de trazabilidad de funcionalidad

La matriz de trazabilidad de funcionalidad tiene el objetivo de comprobar que los requisitos software recogidos en la sección de análisis hayan sido implementados en el sistema y funcionen correctamente.

Para ello, se marcan las intersecciones entre requisitos y pruebas en aquellas pruebas que permiten verificar que el requisito se ha cumplido. No es necesario que el objetivo final de la prueba coincida con el requisito señalado, es suficiente que durante la prueba se demuestre que el requisito se ha cumplido. Siempre y cuando todas las filas de la matriz tengan al menos una marca, se podrá asegurar que las pruebas han validado la recogida del requisito en el sistema. En caso contrario, debería hacerse al menos una prueba específica para cada requisito no comprobado. Por extensión no se han incluido todas las pruebas realizadas, pero sí las más características y que en conjunto no dejen ningún requisitos sin verificar.

	PF_01	PF_02	PF_03	PF_04	PF_05	PF_06	PF_07	PF_08	PF_09	PF_10	PF_11	PF_12	PF_13	PF_14
RF-01	X													
RF-02				X										
RF-03					X									
RF-04			X											
RF-05		X												
RF-06						X								
RF-07							X							
RF-08								X						
RF-09								X						
RF-10									X					
RF-11										X				
RF-12											X			
RF-13												X		
RF-14												X		
RF-15													X	
RF-16													X	
RF-17														X
RNF-01	X													
RNF-02				X										
RNF-03		X		X										
RNF-04					X									
RNF-05						X								
RNF-06			X											
RNF-07			X											
RNF-08			X											
RNF-09			X											
RNF-10		X												
RNF-11							X							
RNF-12								X						
RNF-13								X						
RNF-14								X						
RNF-15									X					
RNF-16										X				
RNF-17											X			
RNF-18													X	
RNF-19														X
RNF-20														X
RNF-21												X		

Tabla 148. Matriz trazabilidad funcionalidad

Capítulo 4.

Planificación, Presupuesto y Recuperación de la Inversión

En este capítulo se detalla la planificación que se seguirá a lo largo de las distintas fases del proyecto, así como el presupuesto del mismo. La planificación se acompaña de un diagrama de Gantt a modo de resumen gráfico del tiempo empleado en cada tarea. El presupuesto incluirá un desglose de todos los costes por separado para tener una mejor apreciación de los costes finales. Además, se realizará una estimación de cuánto tiempo sería necesario para recuperar la inversión inicial.

4.1. Planificación

En este apartado se detalla el seguimiento del presente trabajo fin de grado. En primer lugar, se establece la forma de seguimiento que se va a realizar para, a continuación, especificar la planificación inicial del proyecto, especificando las tareas realizadas y sus plazos. El apartado finaliza con unas líneas en las que se indica la fidelidad respecto de la planificación inicial.

4.1.1. Forma de seguimiento

La forma de seguimiento del trabajo fin de grado ha consistido en reuniones quincenales con el tutor del proyecto. Las reuniones se han mantenido durante los 4 meses del proyecto, desde Mayo a Septiembre de 2014. En estas reuniones se exponían los avances y se establecían los hitos a cumplir para la siguiente reunión. Por cuestiones técnicas y personales, estas reuniones se han podido adelantar o retrasar, aunque nunca se han alargado más de 1 mes.

4.1.2. Planificación inicial

La planificación del trabajo fin de grado se ha formalizado mediante un diagrama de Gantt en el que se especifican las tareas realizadas y las fechas de inicio y de fin. La planificación se ha realizado teniendo en cuenta la ejecución completa del proyecto, pasando por todas sus fases. La siguiente ilustración muestra el diagrama en cuestión:

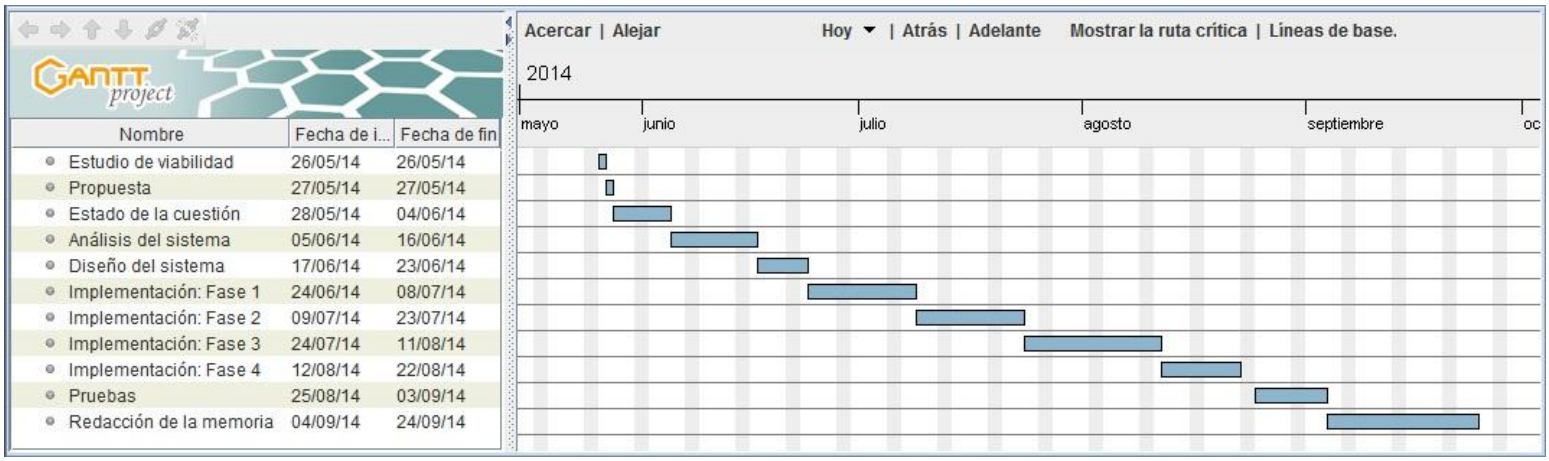

Ilustración 30. Diagrama de Gantt

A continuación se describen en qué han consistido cada una de las tareas mostradas en la ilustración 29:

1. **Estudio de viabilidad:** se estudió si la propuesta era viable para realizar el proyecto antes de proponérselo al tutor.
2. **Propuesta:** se redactó y envió la propuesta del proyecto al tutor.
3. **Estado de la cuestión:** se establece en el contexto en el que se sitúa. En última instancia, se realiza una comparativa entre tecnologías para escoger la más adecuada.

4. **Análisis del sistema:** se describen las características del problema a resolver mediante la definición de los requisitos del sistema
5. **Diseño del sistema**: se define la interfaz del sistema, definiendo todos los componentes que se desean desarrollar en el entorno de acuerdo a los requisitos a cubrir.
6. **Implementación: Fase 1:** Se desarrollan aquellos componentes visuales, definidos en la fase de diseño, correspondientes a la vista detallada del entorno, es decir, los gráficos. En concreto, se implementan únicamente las interfaces de forma estática.
7. **Implementación: Fase 2:** Se desarrollan todas las interacciones entre las pantallas y se implementan todas las animaciones de la aplicación.
8. **Implementación: Fase 3:** Se implementan todas las funcionalidades de una partida.
9. **Implementación: Fase 4:** Se implementa la funcionalidad de selección de carta según el nivel de dificultad
10. **Pruebas**: Se realizan todas las pruebas de la aplicación.
11. **Redacción de la memoria:** Se elabora el presente documento, correspondiente a la documentación del proyecto completo.

4.1.3. Planificación final

El presente trabajo fin de grado ha cumplido la planificación inicial, habiéndose finalizado y entregado en fecha, cumpliendo así todos los objetivos propuestos.

4.2. Presupuesto

En esta sección se detalla el presupuesto estimado para la realización del proyecto, desglosando los costes en función de su naturaleza.

Todos los costes se calculan sin I.V.A., este se aplica en el resumen de costes totales del proyecto que se muestra tras el cálculo detallado de cada parte.

4.2.1. Resumen de tiempo dedicado

El diagrama de Gantt desarrollado en el apartado anterior posibilita el cálculo del número de horas totales dedicadas al proyecto.

A continuación se muestra en detalle el cálculo de horas dedicadas por separado a cada tarea, y la suma en conjunto a la realización de todas ellas. No se cuentan ni los sábados ni los domingos a la hora de realizar el cómputo de los días debido a que no se trabajan.

Estudio de viabilidad: 1 días x 8 horas = 8 horas.
Propuesta: 1 días x 8 horas = 8 horas.
Estado de la cuestión: 6 días x 8 horas = 48 horas.
Análisis del sistema: 8 días x 8 horas = 64 horas.
Diseño del sistema: 5 días x 8 horas = 40 horas.
Implementación: Fase 1: 11 días x 8 horas = 88 horas.
Implementación: Fase 2: 11 días x 8 horas = 88 horas.
Implementación: Fase 3: 13 días x 8 horas = 104 horas.
Implementación: Fase 4: 9 días x 8 horas = 72 horas.
Pruebas: 8 días x 8 horas = 64 horas.
Redacción de la memoria: 15 días x 8 horas = 120 horas.

Tras realizar los cálculos, se comprueba que el tiempo total dedicado al proyecto es de 88 días, que se traducen en 704 horas.

4.2.2. Desglose: Coste de personal

A la hora de calcular el coste, realizaremos una media de lo que cobran los distintos cargos en la actualidad [11]. Se contará que se trabajan 8 horas diarias y 21 días cada mes para realizar los cálculos.

Cargo	*neto/mes (€)*	*Neto/hora (€)*	*Fases*	*Coste (€)*
Analista	2.016	12	Estudio de viabilidad	96
			Propuesta	96
			Estado de la cuestión	576
			Análisis del sistema	768
			Redacción de la memoria	1.440
Total				**2.976**

Diseñador	1.680	10	Diseño del sistema	400
Total				**400**
Programador	1.176	7	Implementación: Fase 1	616
			Implementación: Fase 2	616
			Implementación: Fase 3	832
			Implementación: Fase 4	504
			Pruebas	448
Total				**3.016**
Coste Final				**6.392 €**

Tabla 149. Coste de personal

4.2.3. Desglose: Coste de hardware

Para el desarrollo del proyecto, ha sido necesaria la adquisición de un terminal portátil que servirá para el desarrollo de cada una de las fases del proyecto y un Smartphone para la realización de todas las pruebas de funcionamiento. A continuación se detallan los datos de ambos dispositivos, para el cual se ha calculado el coste imputable a partir de la fórmula de amortización siguiente:

$$\frac{A}{B} * C * D$$

Dónde:

A: nº de meses desde la fecha de facturación en que el equipo es utilizado.
B: periodo de depreciación (60 meses para el portátil y 18 meses para el Smartphone).
C: coste del equipo (sin I.V.A).
D: % del uso que se dedica al proyecto.

Descripción	*Coste (€)*	*% Uso dedicado al proyecto*	*Dedicación (meses)*	*Periodo de deprecación*	*Coste Imputable (€)*
Ordenador portátil (ASUSPRO ADVANCED BU400A)	850	100	4	60	56,67
Sony Xperia U	199	100	4	18	44,22
Coste Total					**100,89**

Tabla 150. Costes de hardware

4.2.4. Desglose: Coste de software

A pesar de que la mayor parte del software con el que se ha trabajado es de código libre, y por tanto no tienen coste alguno, ha sido necesaria la adquisición de diversas licencias software para el desarrollo de ciertas fases del proyecto.

A continuación se detallan las licencias y el coste sin I.V.A. asociado.

Descripción	*Cantidad*	*Coste (€)*	*Amortización (años)*	*Duración del proyecto (meses)*	*Coste Imputable (€)*
Microsoft Office Professional 2010	1	499	1	4	166,33
Windows 8	1	180	5	4	12
Adobe Photoshop CSS	1	250	5	4	16,67
Coste Total					**195**

Tabla 151. Costes de software

4.2.5. Resumen de costes

Tras analizar los distintos costes asociados al proyecto, se resume a continuación cada uno de ellos y se establece el coste total del proyecto. A la suma de costes calculada, se añade un 20% de costes indirectos para cubrir los riesgos del proyecto y los gastos que no han sido tenidos en cuenta al realizar el presupuesto.

Se detalla el coste total con y sin I.V.A.

Descripción de Costes	*Coste Total (€)*
Personal	6.392
Costes de hardware	100,89
Costes de software	195
Costes indirectos (20%)	1.337,58
Total sin I.V.A	8.025,47
Total con I.V.A (21%)	***9.710,82***

Tabla 152. Costes Totales

El coste total del proyecto sin I.V.A asciende a la cantidad de 8.025,47 € (OCHO MIL VEINTICINCO EUROS CON CUARENTA Y SIETE CÉNTIMOS) y aplicando el 21% de I.V.A correspondiente, el precio final del proyecto es de 9.710,82 € (NUEVE MIL SETECIENTOS DIEZ EUROS CON OCHENTA Y DOS CÉNTIMOS).

4.3. Recuperación de la inversión

Dado que el proyecto es una aplicación móvil que puede generar beneficios al ser subidas a Google Play, vamos a realizar una estimación de qué necesitaríamos para empezar a recibir ingresos a partir de ésta aplicación. Para ello vamos a analizar los diversos modelos de negocio que existen a la hora de subir aplicaciones [12][13]. Existen muchos modelos de negocio, consultaremos los cuatro más comunes y utilizados que generen ingresos, haciendo especial hincapié en el modelo publicitario.

4.3.1. Modelos de negocio

En este apartado estimaremos los ingresos con respecto a los cuatro modelos de negocio más utilizados que generen ingresos, los cálculos del modelo publicitario los veremos en el siguiente apartado debido a su complejidad.

4.3.1.1. App de pago

Se trata de una aplicación de pago. Generalmente se suele poner el precio de la aplicación entre 1 y 5 euros por descarga, de los cuales el 70% de los ingresos es para el programador y el 30% para Google.

Estimamos que se pondría la aplicación al precio de 0,99 € para realizar los cálculos que suele ser el precio más utilizado en este tipo de aplicaciones. Por cada aplicación que se descargase, recibiríamos el 70% de 0,99 € = 0,693 €. Al ser los gastos de la aplicación de 9.710,82 €, necesitaríamos un total de 14.013 para cubrir el coste del proyecto.

4.3.1.2. App gratuita con publicidad

La aplicación se descarga gratuitamente y genera rentabilidad mediante banners de publicidad de diferentes tipos que veremos en el apartado 4.3.2.

4.3.1.3. Fremium

Este modelo ofrece una versión gratuita de la aplicación, completamente funcional, pero con ciertas limitaciones y una versión de pago eliminando dichas limitaciones de la anterior. Suelen ser aplicaciones que generan el dinero mediante compra de la funcionalidad dentro de la misma aplicación (In App Purchase, IAP).

Para estimar este cálculo, utilizamos que la versión no limitada costaría 0,99 € pero no habría que pagar el 30% a Google Al ser los gastos de la aplicación de 9.710,82 €, necesitaríamos que 9809 usuarios comprasen la versión no limitada para poder cubrir los gastos del proyecto.

El porcentaje de descargas en este modelo de negocio que compran la versión no limitada es del 1%, por lo que harían falta 980900 descargas aproximadamente de la versión gratuita para poder recuperar la inversión del proyecto.

4.3.1.4. Demostración

Una aplicación de demostración es aquella que muestra un ejemplo del producto y te invita a comprar la versión completa si te ha gustado. Suele ser un tiempo durante el cual puedes poseer la aplicación y después que necesites comprarla para seguir usándola.

Estimamos que se pondría la aplicación al precio de 0,99 € después de usarse quince días. Los cálculos serían igual que en el apartado 4.3.1.1., con la excepción de que para recuperar la inversión, en vez de necesitar 14.013 usuarios que se descarguen la aplicación, se necesitarían 14.013 usuarios que compren la aplicación después del periodo de prueba.

4.3.2. Modelo publicitario

En este modelo intervienen 4 roles:

- Programador: Crea la aplicación y reserva un espacio para incluir los anuncios publicitarios.

- Anunciante: Una empresa o usuario paga una cierta cantidad por anunciar su producto en varias aplicaciones para conseguir popularidad.

- Intermediario: sirve anuncios en las aplicaciones. Hace posible la publicidad y cobra un pequeño porcentaje a cambio cuando se efectúa una acción.

- Usuario final: el cliente de la aplicación. Interactúa de una determinada forma conlos anuncios generando ganancias para el programador y una visita, descarga o venta para el anunciante.

La interacción se produce entre el usuario final de la aplicación y el anuncio. Existen varias formas de interacción, las más comunes son las siguientes cuatro:

4.3.2.1. CPM

Coste por cada mil impresiones: es la más pasiva de las cuatro. Cuando mil usuarios finales visualizan el anuncio, el programador recibe un pequeño ingreso.

Estimamos que por cada mil impresiones se cobrarían 0,02 € para realizar los cálculos. Al ser los gastos de la aplicación de 9.710,82 €, necesitaríamos un total de 485.541.000 impresiones de publicidad por pantalla para cubrir el coste del proyecto.

4.3.2.2. CPC

Coste por click: se trata de realizar un ingreso cuando un usuario hace click sobre el anuncio. El ingreso percibido es superior al del CPM.

Estimamos que por cada click se cobrarían 0,10 € para realizar los cálculos. Al ser los gastos de la aplicación de 9.710,82 €, necesitaríamos un total de 97.109 clicks en la publicidad para cubrir el coste del proyecto.

4.3.2.3. CPL

Coste por "lead": en este caso el ingreso se percibe cuando el usuario final hace click en el anuncio y rellena un formulario, se registra o se da de alta. Por este método se cobra más que los anteriores, pero es más costoso.

Estimamos que por cada formulario rellenado, registro o alta se cobrarían 2 € para realizar los cálculos. Al ser los gastos de la aplicación de 9.710,82 €, necesitaríamos un total de 4.856 interacciones para cubrir el coste del proyecto.

4.3.2.4. CPA

Coste por acción: el ingreso se recibe cuando el usuario final realiza una acción determinada, generalmente una compra, una descarga o similar. Suelen ser ingresos bastante altos, usualmente comisiones del producto vendido.

Estimamos que por cada acción se cobrarían 5 € para realizar los cálculos. Al ser los gastos de la aplicación de 9.710,82 €, necesitaríamos un total de 1.943 acciones para cubrir el coste del proyecto.

4.3.3. Conclusiones de los modelos de negocio

Una vez vistos los modelos de negocio principales, basándonos en sus características, se toma la opción de que para la aplicación Cuatrola Pro lo mejor sería hacer un modelo de negocio híbrido entre el modelo de negocio fremium y el publicitario.

Se podría hacer que las descargas fuesen gratuitas y que se pudiese acceder a todas las funcionalidades de la aplicación excepto a poder jugar en el nivel difícil. Esta versión gratuita contendría publicidad. Analizando los diferentes tipos de modelo publicitario, se elegiría el CPC (coste por click), puesto que aunque no es el que genera ingresos más altos, es el más rentable en relación esfuerzo del usuario final e ingresos.

La versión Premium podría ponerse a 0,99 € y tendría de diferencia con la versión gratuita que se podría jugar en el nivel difícil y que no contendría publicidad.

Debido a que en la opción fremium el volumen de descargas es un factor vital para la aplicación, podrían usarse otros modelos de negocio a la vez como el de demostración, dando la posibilidad a un usuario final de invitar a cinco amigos a que se descarguen la aplicación y si lo hacen que puedan tener la versión Premium un periodo limitado de días.

Capítulo 5.

Marco regulador

En este apartado, considerando que Cuatrola Pro es una aplicación que no está subida a Google Play y que no genera beneficios, no tiene que acogerse a ninguna ley.

Si por el contrario en un futuro se desease que fuese subida a Google Play y generase ingresos habría que darse de alta sólo en Hacienda y declarar los ingresos recibidos a través de esta actividad a través del IRPF.

En caso de que estos ingresos fueran de cantidades pequeñas se podría hacer a través de la regularización de ingresos a través del IRPF [14], mientras que si éstos son más elevados deberíamos hacerlo a través de las declaraciones trimestrales con el modelo 130.

Chapter 6.

Conclusions

This chapter summarizes the main contributions of this project, lists the problems encountered along its execution and establishes recommendations for future work. To conclude, different personal opinions related with the performance of this work are presented.

6.1. Contributions

Currently, the game Cuatrola is a game that is not very popular across Spain, usually played only in Madrid and Extremadura. This is the reason why no developer has delved deeper into the creation of applications to play this game and any player Cuatrola not have the ability to play competently on your mobile device.

Today, the only operating system that has some application of the game Cuatrola is Android. This application has many shortcomings and limitations of features that make playing this game is made very heavy and monotonous. The plus point of this application is that it has a good interface and graphics that has served to perform the interface intuitive of Cuatrola Pro.

Another advantage of the application is able to meet the estimated number of players of Cuatrola due to downloads of the application, to be able to estimate the revenue and downloads that might have to upload it to Google Play.

Finally, many of the features of the application of Cuatrola have improved and added thanks to the experience of the users of the application because it has a large number of critical commentaries on the failures and shortcomings of the functionalities.

6.2. Future works

The future works that are to be developed for this application are based on getting three important characteristics:

1. **Multiplatform:** aims to develop, in a future, the application for the operating systems most used by users. Today, these operating systems are iOS and Windows Phone.
2. **Interactivity:** This application is developed to play offline with participants guided by the algorithms implemented in the application. In the future we want to implement a version to play with other players of the application online possessing a common database in the cloud where everyone can view their own stats and the stats of their opponents to create more competition.
3. **Profitability:** once the online version is generated, we want to create a game system based on tokens. Initially a limited number of tokens, which always increases or descending according to the games won will be given. Once a player runs out of chips, he has to wait a time while to pick up a few or he cans get with real money, getting benefits application.

6.3. Problems encountered

The fundamental problems encountered revolve around technology, despite being selected, has led several complications.

One complication was the initial lack of technology, which has led to make progress slower during the project and the planning when we have made the implementation has been made on the rise.

Another problem has been that, during the game, many of the actions are managed through animations. Android still has limitations on how animations are concerned and has been very expensive to get the final result.

Finally, the third problem has been the different densities of screen on Android devices. This has caused problems with the images and animations with respect to size and position on the screen, when density changes.

Because it is a project of four months, the above problems have generated that have to make a planning in which has not had time to solve the problem of density display and therefore has not been able to upload the application to Google Play.

6.4. Personal opinions

The development of this final thesis has been an overall satisfaction in all aspects. In the first place, because it supposes the last door to cross to finish the degree, but above all it supposes a professional and personal satisfaction. In the previous section it has already mentioned the few knowledge of Android technology, so that the knowledge acquired during this thesis have been very large, especially considering that Android technology is basic element (among many others) for a good informatics engineering.

Another of the satisfactions of this project has been to generate my first Android application. I have had interviews to date and show this application increases my chances of entering the job over other candidates

In addition, this project has put into practice the knowledge of various subjects, having done a project completely, through all its phases. This also reveals the difficulty that supposed and the fundamental importance of planning tasks and follows all carefully: not everything is to develop, but also fully seat bases, defining what it wants to develop, why, etc. It is also learned the importance of tasks with time instead of leaving everything to the last minute. Given the scale of a final project, and comparing the practices of the subjects (less difficult and made in smaller team), in the first instant it is found that this requires a planned and consistent work. To complement the talk about architectures, one can generalize this: it is essential to expand and diversify your skills, so that they know how to analyse problems and select the best tools for their solution.

One of the things that surprised me during this project has been the ability of implicit synthesis of the human brain, because when I played this game, make decisions of whether you play a special play or which card to throw in each turn take an average time of one second, become almost automatic. When performing algorithms, I found it expensive to transform this implicit synthesis in a methodical algorithm because in the elections there are a large number of unknowns are correlated. It also has the difficulty that there are times during the game that knows the cards what other participants carry, but there are others that are based on probabilities because the information is not known. All this information is taken into account in the brain and, apparently without a methodical algorithm returns the solution in a second.

In summary, and to conclude this section, the overall satisfaction achieved is reiterated: it is gratifying to see how starting from virtually "nothing", that is, some ideas not completely defined, it gets a whole development of a tangible project.

Anexo I

Reglas de la Cuatrola

Debido a que el juego de la Cuatrola posee unas reglas para ser jugado y no es popularmente conocido, en este anexo se explica toda la información necesaria para que el lector pueda comprender cómo se juega a la Cuatrola.

Información general

La Cuatrola es un juego de cartas en el que 2 parejas de jugadores compiten entre sí. La disposición de los jugadores sobre el tapete o tablero de juego es en cruz, tal que los componentes de una misma pareja estén cara a cara.

De las 48 ó 50 cartas de la baraja española, sólo se utilizan la sota o 10, el caballo o 11, el rey o 12, el tres o 3 y el as o 1, de los cuatro palos (oros, copas, espadas y bastos).

Una partida de Cuatrola tiene muchas manos y cada mano tiene cinco bazas.

Origen

Este juego, de la familia del tute, con arraigo en Extremadura y algunas zonas de Madrid, aunque se puede encontrar gente que sabe jugar en muchos puntos de España, debido mayormente a la emigración.

Inicio

Una partida se inicia tirando una carta boca arriba en el centro del tapete, según el palo de esta carta comenzará a repartir un participante. Si el palo de la carta es oros, repartirá el jugador de la derecha del jugador; si es copas, comenzará a repartir el participante que se encuentra enfrente del jugador de la partida; si es espadas, comenzará a repartir el participante situado a la izquierda del jugador de la partida; y si es bastos comenzará a repartir el jugador de la partida.

Aquel participante que reparta, repartirá las veinte cartas al azar entre los cuatro participantes, empezando por el que tenga a su derecha y siguiendo un sentido anti horario.

El Triunfo es el palo de la última carta repartida, por tanto, el participante que reparta deberá mostrar claramente al resto de los participantes la última carta que reciba.

Acto seguido, los participantes han de declarar su intención de jugarse una Quintola, una Cuatrola, un solo o no jugarse nada según las cartas que le hayan sido asignadas y según el riesgo que quiera correr. Para ello, se realizará un orden de declaración que empieza con el jugador que abre la mano, siguiendo el sentido anti horario, y acaba con el jugador que repartió.

Si algún participante se juega alguna jugada especial, su pareja no participa en esa mano y comienza la mano y la primera baza de ésta.

Reglas de asistencia

Cuando se abre la baza se puede tirar cualquier carta de las 5 que posea el participante en la mano. Si no se abre la baza se siguen las reglas explicadas a continuación:

Se entiende por arrastre la abertura de una baza con una carta del mismo palo que el triunfo. Se debe montar una carta que abra una baza, incluido un arrastre. Es decir, se debe tirar una carta del mismo palo que la primera tirada en esa baza, y de mayor valor o peso (explicado más adelante.

De no tener cartas para montar, se está obligado a asistir, incluido a un arrastre. Esto quiere decir que se debe tirar una carta del mismo palo que la carta que abre la baza.

De no tener cartas para asistir, se está obligado a fallar. Es decir, se debe tirar una carta del palo del Triunfo. Si ya hay un fallo de otro participante y la baza no es arrastre, se está obligado a montar ese fallo. Si no se tienen cartas para montar un fallo previo, se puede tirar cualquiera de las que posea el jugador (sea triunfo o no).

En caso de existir un fallo previo, tampoco se está obligado a montar la carta que inició la baza, pero sí a asistir.

La baza es ganada por el jugador que hubiera tirado el triunfo de más valor, si lo hubiera. En caso contrario (si no hubiese triunfos en la baza), la baza es ganada por el jugador que hubiera tirado la carta que, siendo del palo que abrió la baza, es de mayor valor.

Reglas de puntuación

Las reglas de puntuación de Madrid y Extremadura son diferentes, por lo que se explicarán por separado.

Extremadura

En manos sin apuesta y en manos en que hay un Solo en juego:

- Los ases valen 11 puntos; los treses valen 10 puntos; los reyes valen 4 puntos; los caballos valen 3 puntos; y las sotas valen 3 puntos.
- La pareja que consiga la última mano recibe 10 puntos extras.
- Los cantes valen 20 puntos extra si no son del palo del triunfo y 40 puntos extras si son del palo del triunfo.

Una partida sin apuesta representa 1 punto de partida para la pareja que gane. Un solo son 2 puntos de partida para la pareja ganadora. Una Cuatrola son 4 puntos de partida. Una Quintola representa 5 puntos de partida.

La partida es ganada por la pareja que consigue antes 21 puntos de partida.

Madrid

En manos sin apuesta y en manos en que hay un Solo en juego:

- Los ases valen 11 puntos; los treses valen 10 puntos; los reyes valen 4 puntos; los caballos valen 3 puntos; y las sotas valen 3 puntos.
- La pareja que consiga la última mano recibe 10 puntos extras.
- Los cantes valen 20 puntos extra si no son del palo del triunfo y 40 puntos extras si son del palo del triunfo.

Una partida sin apuesta representa 1 punto de partida para la pareja que gane si su puntuación no sobrepasa los 100 puntos y 2 puntos de partida si su puntuación sobrepasa los 100 puntos. Un solo son 3 puntos de partida para la pareja ganadora. Una Cuatrola son 4 puntos de partida. Una Quintola representa 5 puntos de partida.

La partida es ganada por la pareja que consigue antes 21 puntos de partida.

Solos, Cuatrolas y Quintolas

Tras el reparto inicial, un participante puede apostarse un Solo si cree que puede ganar la mano sin ayuda de su pareja.

Si, en cambio, apuesta una Quintola, debe ganar las 5 bazas de que consta la mano.

Si se apuesta una Cuatrola debe ganar al menos 4 bazas de esa mano.

En Cuatrolas y Quintolas la puntuación y los cantes no tienen valor puesto que se gana la mano según las bazas ganadas, y no por los puntos obtenidos.

En Solos y manos sin apuesta, la mano es ganada por la pareja que sume más puntos conforme al valor de las cartas de las bazas que ganó, así como de los cantes y de los 10 puntos extras recibidos por llevarse la última baza (conocidos como “las 10 de monte”).

Cantes

Siempre que se cumplan las condiciones que se explican más adelante, un cante supone la suma de 20 puntos extra si se posee el caballo y el rey de un mismo palo, o de 40 si el palo se trata del triunfo. En ese caso, el participante que posee el cante debe indicar el cante que posee.

Las condiciones que deben cumplirse para poder cantar son:

- Tener el caballo y el rey del mismo palo en el momento que el participante que posee el cante o su pareja, ganan la baza. Así, si gana la pareja adversaria, no puedes cantar nada. Cuando acaba la baza ganada por el poseedor del cante o su pareja, tiene que tener en las cartas que conserva en la mano dicho caballo y rey.

- No estar en juego una Cuatrola ni una Quintola.
- Estar jugando (o sea, que la pareja del jugador poseedor del cante no esté jugándose ninguna jugada especial).
- No haber cantado ya en ese palo.
- No haber cantado el poseedor del cante ya otro cante en la misma baza. Un participante sólo podrá realizar un cante por baza en el caso de poseer más de uno.

Anexo II.

Manual de usuario

Este anexo tiene como objetivo mostrar en detalle cómo desenvolverse con la aplicación "Cuatrola Pro". Para ello, se establecerán una serie de pasos a seguir para aprender a navegar por la interfaz a modo de tutorial.

PASO 1. PANTALLA PRINCIPAL

Es la primera pantalla de la aplicación. En ella se muestran dos botones.

- ✓ *Botón Jugar*
  Una vez que pulsas este botón, se produce el **PASO 3** de este manual
- ✓ *Botón Reglas*
  Una vez que pulsas este botón, se produce el **PASO 2** de este manual

PASO 2. PANTALLA REGLAS

En esta pantalla aparecen las reglas del juego Cuatrola Pro, las cuales puedes ir leyendo arrastrando el dedo de abajo hacia arriba de la pantalla para que se vayan mostrando. También aparece un botón.

- ✓ *Botón Volver*
  Al ser presionado, se retorna al **PASO 1** de este manual.

PASO 3. ACCESO JUGAR

Se produce cuando se ha pulsado el botón jugar de la pantalla principal de la aplicación. En este punto pueden sucederse un par de situaciones.

- ✓ *Que no existan aún jugadores creados en la aplicación*
  Cuando no existen jugadores creados en la aplicación, aparece el **PASO 4** de este manual.
- ✓ *Que existan jugadores creados en la aplicación*
  Cuando existen jugadores creados en la aplicación, aparece el **PASO 5** de este manual.

PASO 4. PANTALLA CREAR JUGADOR

En esta pantalla aparece un cuadro de texto en el que hay que introducir el nombre que queremos crear en la base de datos de la aplicación para jugar. También aparece un botón.

- ✓ *Botón Crear Jugador*
  Cuando es presionado, comprueba que el nombre introducido en el cuadro de texto de la pantalla no esté vacío ni esté ya creado en la base de datos de la aplicación. Si se diese alguna de estas circunstancias, mostraría un mensaje informativo por pantalla. Si no se dan, creará al jugador en la base de datos y pasará al **PASO 9** de este manual.

PASO 5. PANTALLA SELECCIONAR JUGADOR

En esta pantalla aparece una lista desplegable con todos los nombres de los jugadores creados en la aplicación. Se puede seleccionar cualquier jugador creado a través de pulsar dicha lista. También aparecen tres botones.

- ✓ *Botón Estadísticas*
  Cuando es presionado, pasará al **PASO 6** de este manual pasándole como información el nombre del jugador que haya seleccionado en ese momento en la lista desplegable.
- ✓ *Botón Jugador Nuevo*
  Cuando es presionado, pasará al **PASO 4** de este manual.
- ✓ *Botón Seleccionar Jugador*
  Cuando es presionado, pasará al **PASO 7** de este manual.

PASO 6. PANTALLA ESTADÍSTICAS

En esta pantalla se muestran las estadísticas del jugador que se han recibido al ser llamada por niveles de dificultad. Según el nivel de dificultad mostrado, poseemos tres botones.

- ✓ *Botón Siguiente*
  Este botón se muestra en las estadísticas del nivel fácil y del nivel medio. Cuando es presionado, muestra las estadísticas del nivel superior de dificultad del que se encuentre.
- ✓ *Botón Anterior*
  Este botón se muestra en las estadísticas del nivel medio y del nivel difícil. Cuando es presionado, muestra las estadísticas del nivel inferior de dificultad del que se encuentre.
- ✓ *Botón Volver*
  Este botón se muestra en las estadísticas del nivel difícil. Cuando es presionado, pasará al **PASO 5** de este manual.

PASO 7. COMPROBAR REANUDAR PARTIDA

Se produce cuando se presiona el botón Seleccionar Jugador de la pantalla Seleccionar Jugador. Comprueba si el jugador seleccionado tiene una partida empezada e inacabada. Según esta comprobación, se pueden dar dos situaciones.

- ✓ *Jugador Seleccionado posee partida inacabada*
  Si posee partida inacabada el jugador que hay seleccionado, pasará al **PASO 8** de este manual.
- ✓ *Jugador Seleccionado no posee partida inacabada*
  Si no posee partida inacabada el jugador que hay seleccionado, pasará al **PASO 9** de este manual.

PASO 8. PANTALLA REANUDAR PARTIDA

En esta pantalla se pregunta si se quiere continuar la partida que hay inacabada. Posee dos botones.

- ✓ *Botón Si*
  Cuando es pulsado, pasará al **PASO 10** de este manual pasándole toda la información de la partida inacabada del jugador.
- ✓ *Botón No*
  Cuando es presionado, se indica en base de datos que la partida se ha perdido y se pasará al **PASO 9** de este manual.

PASO 9. PANTALLA SELECCIONAR OPCIONES

En esta pantalla se seleccionan las opciones que se desean que tenga la partida a jugar. Está formado por las opciones de seleccionar la comunidad de tanteo, seleccionar el nivel de dificultad, y seleccionar si quieres que tu compañero se pueda jugar jugadas especiales. También está compuesta por un botón.

- ✓ *Seleccionar Comunidad de Tanteo*
  Para realizar esta selección, se cuenta de dos opciones, una Madrid y otra Extremadura de las que por defecto viene seleccionada Madrid. Cuando pulsas una u otra, la otra se deselecciona e indica según las reglas de puntuación de qué comunidad autónoma se quiere jugar la partida.
- ✓ *Seleccionar nivel de dificultad*
  Para elegir qué nivel de dificultad se desea jugar, existen tres botones llamados fácil, medio y difícil. Por defecto no hay ninguno seleccionado. Cuando se selecciona alguno, se deseleccionan los otros y se marca seleccionado el botón pulsado.
- ✓ *Botón Seleccionar Jugador*
  Para decidir si se quiere que el compañero del jugador se pueda jugar jugadas especiales hay un cuadro seleccionable que si está pulsado es que el compañero se puede jugar jugadas especiales y si no, no. Por defecto, aparece el cuadro seleccionable marcado.
- ✓ *Botón Empezar*
  Cuando es pulsado, se comprueba si el nivel de dificultad ha sido seleccionado. Si no está seleccionado se muestra mensaje informativo por pantalla. Si se ha seleccionado, pasará al **PASO 10** de este manual pasándole toda la información de las opciones de la partida seleccionadas.

PASO 10. PANTALLA TAPETE PARTIDA

Esta pantalla es la encargada de gestionar las partidas. Dependiendo de la información recibida genera una partida nueva o carga los datos de una partida que estaba inacabada. Si la información recibida es de una partida inacabada, carga la información de la partida y comienza a repartir la última mano en la que el jugador se salió de la partida. Posteriormente gestiona la partida cómo se explica en el **PASO 11** de este tutorial.

PASO 11. CÓMO JUGAR UNA PARTIDA

Lo primero de todo es ver qué participante de la partida es el que reparte. Si la partida es una que ya se comenzó, repartirá el jugador al que le toque repartir la partida. Si se trata de una partida nueva, se sacará una carta al centro del tapete, según el palo de esta carta repartirá un participante.

- Si sale una carta del palo de oros, repartirá el participante de la derecha.
- Si sale una carta del palo de copas, repartirá el participante de enfrente.
- Si sale una carta del palo de espadas, repartirá el participante de la izquierda.
- Si sale una carta del palo de bastos, repartirá el jugador de la partida situado en la parte inferior de la pantalla.

Una vez que se ha repartido una mano, la última carta repartida será la carta pinte, que se mostrará durante toda la mano en la esquina superior izquierda de la pantalla, y si ha repartido unos de los participantes de la partida que no sea al jugador, se mostrará detrás de las cartas del jugador que reparta hasta la tire en algún turno de su mano.

Luego se comenzará por turnos, en sentido anti horario, comenzando el participante siguiente al jugador que haya repartido.

Primeramente los participantes elegirán que tipo de jugada quieren jugar según las cartas que les haya tocado en la mano. Cuando llegue el turno del jugador de la partida aparecerá la pantalla de elegir jugada que se explica en el **PASO 12**. Si algún participante decida jugarse alguna jugada especial, se comenzará la mano, excluyendo de ésta al compañero del participante que se juega la jugada.

Una vez que comience la mano, se irán rotando los turnos entre los participantes que jueguen la partida. En cada turno, el participante que posea el turno deberá seleccionar una carta a tirar al tapete que se ajuste a las reglas de la Cuatrola. El jugador que gane una baza, será el que posea el primer turno en la siguiente baza.

Si el jugador de la partida posee un cante en la mano cuando su compañero o él ganan una baza, podrá cantar. Para ello tendrá que seleccionar en la lista desplegable situada arriba a la izquierda de la pantalla el palo en el que desea cantar.

A la izquierda de las cartas de la mano del jugador de la partida, aparecerá el botón de Ordenar, cuando el jugador presione este botón, las cartas de su mano se ordenarán por palos y dentro de éstos por puntuación.

Cuando acaba la última mano de cada baza, se realiza el cálculo de puntos de ambas parejas y se actualiza el marcador de la partida según el tanteo de la comunidad autónoma que se haya seleccionado para jugar la partida.

La partida acabará cuando alguna de las dos parejas participantes lleguen o superen los 21 puntos.

PASO 12. PANTALLA ELEGIR JUGADA

En esta pantalla el jugador de la partida seleccionará el tipo de jugada que desea jugar según las cartas que posea en la mano. Si desea jugarse una jugada especial, tendrá que seleccionar las opciones de Solo, Cuatrola o Quintola según decida. Si no desea jugarse ninguna jugada, seleccionará la opción de Nada. En esta pantalla también se muestra un botón.

- ✓ *Botón Opciones*
  Al ser pulsado se pasará al **PASO 13**.

PASO 13. PANTALLA OPCIONES PARTIDA

Esta pantalla muestra las opciones dentro de una partida. Aparecen tres opciones dentro de dicha pantalla.

- ✓ *Opción Cambiar Nombres*
  Al ser pulsada se pasará al **PASO 14**.
- ✓ *Opción Continuar Partida*
  Al ser pulsada se pasará al **PASO 12**.
- ✓ *Opción Abandonar*
  Al ser pulsada se pasará al **PASO 15**.

PASO 14. PANTALLA CAMBIAR NOMBRES

Esta pantalla se realiza el cambio de nombres de los participantes de la partida. Consta de tres cuadros de texto en los que hay que introducir los nombres de los participantes que se desean cambiar en la base de datos de la aplicación para jugar contra ellos en la partida. Además tiene un botón.

- ✓ *Botón Cambiar Nombres*
  Al ser pulsado comprueba que todos los nombres de los participantes de la partida, incluido el jugador de la partida, sean diferentes. Si alguno coincide, se mostrará un mensaje informativo para que no se introduzca ningún nombre repetido. Si ninguno se repite, se pasará al **PASO 12**.

PASO 15. PANTALLA ABANDONAR PARTIDA

En esta pantalla se pregunta si se quiere abandonar la partida en juego dándose ésta por perdida. Posee dos botones.

- ✓ *Botón Si*
  Cuando es pulsado, se indica en base de datos que la partida se ha perdido y se pasará al **PASO 1** de este manual.
- ✓ *Botón No*
  Cuando es presionado, se pasará al **PASO 12** de este manual.

Bibliografía

[1] Reglas de la cuatrola

http://es.wikipedia.org/wiki/Cuatrola

[2] Evolución de los juegos.

http://elrincondelasmaquinitas.blogspot.com.es/2014/02/la-evolucion-de-los-juegos-en-telefonos.html

[3] Comparativa del número de aplicaciones de las plataformas móviles.

http://www.movilzona.es/2014/08/08/google-play-supera-por-primera-vez-a-la-app-store-de-apple-en-numero-de-aplicaciones/#comments

[4] Crecimiento de los dispositivos Android

http://www.elandroidelibre.com/2014/01/android-alcanza-los-dos-mil-millones-de-dispositivos-casi-mil-millones-solo-en-2013.html

[5] Versiones actuales de iOS

http://www.applesfera.com/ios/camino-de-ios-7-1-2-tal-vez-la-ultima-actualizacion-de-ios-7-antes-de-ios-8

[6] Características Windows Phone 8.1

http://www.xatakamovil.com/sistemas-operativos/windows-phone-8-1-toda-la-informacion

[7] “Statista: Google überholt Apple,” [Online]. Available:

http://de.statista.com/themen/882/apps-app-stores/infografik/810/anzahl-der-verfuegbaren-apps-in-den-top-app-stores/.

[8] “International Standard ISO/IEC 12207. Software Life Cycle Processes,” [Online]. Available:

http://www.abelia.com/docs/12207cpt.pdf.

[9] “Desarrollo de Software Iterativo e incremental,” 14 febrero 2007. [Online]. Available:

http://fernandosoriano.com.ar/?p=14.

[10] R. S. M. M. Armando Cabrera, “Procesos de Ingeniería del Software,” [Online]. Available:

http://es.slideshare.net/rfsolano/procesos-de-ingenieria-del-software.

[11] Herramienta cálculo de salarios medios InfoJobs Trends Salarios

http://salarios.inforjobs.net/

[12] Modelos de negocio de aplicaciones Android

http://www.emezeta.com/articulos/conferencia-ganar-dinero-con-android.

[13] Modelos de negocio de aplicaciones Android

http://www.lamotora.com/index.php/blog/166-monetizacion-apps-moviles.

[14] Regularización de ingresos según el IRPF

http://www.pymesyautonomos.com/fiscalidad-y-contabilidad/regularizar-ingresos-a-traves-del-irpf.